Cisco IE3500 Series Switch Software Configuration Guide, Cisco IOS XE 17.17.1

First Published: 2025-07-31

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

Preface

This preface describes the conventions of this document and information on how to obtain other documentation. It also provides information on what's new in Cisco product documentation.

Document Conventions

This document uses the following conventions:

Convention	Description
^ or Ctrl	Both the ^ symbol and Ctrl represent the Control (Ctrl) key on a keyboard. For example, the key combination **^D** or **Ctrl-D** means that you hold down the Control key while you press the D key. (Keys are indicated in capital letters but are not case sensitive.)
bold font	Commands and keywords and user-entered text appear in **bold** font.
Italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.
`Courier` font	Terminal sessions and information the system displays appear in `courier` font.
`Bold Courier` font	**`Bold Courier`** font indicates text that the user must enter.
[x]	Elements in square brackets are optional.
...	An ellipsis (three consecutive nonbolded periods without spaces) after a syntax element indicates that the element can be repeated.
|	A vertical line, called a pipe, indicates a choice within a set of keywords or arguments.
[x | y]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
{x | y}	Required alternative keywords are grouped in braces and separated by vertical bars.

Convention	Description
[x {y | z}]	Nested set of square brackets or braces indicate optional or required choices within optional or required elements. Braces and a vertical bar within square brackets indicate a required choice within an optional element.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
< >	Nonprinting characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Reader Alert Conventions

This document may use the following conventions for reader alerts:

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the manual.

Tip Means *the following information will help you solve a problem.*

Caution Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Timesaver Means *the described action saves time.* You can save time by performing the action described in the paragraph.

Take note of the following general safety warnings:

Warning

Statement 1071—Warning Definition

IMPORTANT SAFETY INSTRUCTIONS

Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Read the installation instructions before using, installing, or connecting the system to the power source. Use the statement number at the beginning of each warning statement to locate its translation in the translated safety warnings for this device.

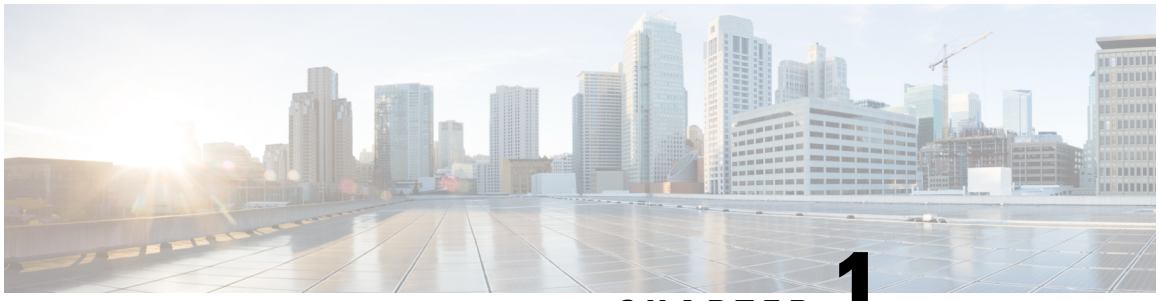
SAVE THESE INSTRUCTIONS

Communications, Services, and Additional Information

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business results you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit Cisco DevNet.
- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.

Cisco Bug Search Tool

Cisco Bug Search Tool (BST) is a web-based tool that acts as a gateway to the Cisco bug tracking system that maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. BST provides you with detailed defect information about your products and software.



CHAPTER 1

Overview

Overview of the Cisco IE3500 Series Switch

Cisco IE3500 Rugged Series Switches

These are ruggedized switching platforms and provides superior high-bandwidth switching and proven Cisco IOS XE Software for industrial environments.

These DIN-rail Industrial Ethernet switches are designed to cater deployments where hardened products are required, including factory automation, smart cities, energy and process control, Intelligent Transportation Systems (ITS), energy production sites, smart city programs, and mining. They bring improved overall performance, greater bandwidth, a richer feature set, enhanced hardware, and class leading security

The switches include security features to provide secured networking environment

- Cisco Trusted Platform Module (TPM)—serves as a hardware root-of-trust for secure boot.
- Secure Boot—uses a public key to validate each subsequent booting stage.
- Chip guard—Cisco developed security feature that records unique ID of critical system to prevent hardware tampering.

Cisco IE3500H Heavy Duty Series Switches

These are the next-generation managed IP67 PoE switches powered by Cisco IOS XE and are ideal for deployment in the harshest environments. They are IP67-rated for water and dust resistance and are hardened to withstand temperatures ranging from freezing cold to extreme heat (-40 to 85°C), as well as severe shock and vibration.

The switches are available with up to 24 ports, offering either Fast Ethernet or all Gigabit Ethernet with M12 connectors. These switches can be wall-mounted and deployed without a housing cabinet, offering a power budget of 360W, and supporting Power over Ethernet (PoE), PoE+, and Universal Power over Ethernet (UPOE) at 60W.

The switches are equipped with advanced network-based security, segmentation, and visibility features for the most demanding industrial environments. They extend the power of intent-based networking to the harshest Internet of Things (IoT) edge, with use cases in industries such as mining, railways, and manufacturing.

Configuring the Switch Using the Web User Interface

This document walks you through the steps to access and configure your switch using the Web UI.

Note Any figures included in the document are shown for illustrative purposes only.

Introduction to Day 0 WebUI Configuration

After you complete the hardware installation, you need to setup the switch with configuration required to enable traffic to pass through the network. On your first day with your new device, you can perform a number of tasks to ensure that your device is online, reachable and easily configured.

The Web User Interface (Web UI) is an embedded GUI-based device-management tool that provides the ability to provision the device, to simplify device deployment and manageability, and to enhance the user experience. You can use WebUI to build configurations, monitor, and troubleshoot the device without having CLI expertise.

Device Configuration

Use the procedures mentioned here to configure the device with basic and advanced settings. Once complete, you can access the device through the WebUI using the management interface IP address.

Connecting to the Switch

Note Before proceeding, ensure you have completed the Express Setup outlined in the hardware installation guide.

Procedure

Step 1 Make sure that no devices are connected to the switch.

Step 2 Connect one end of an ethernet cable to the switch and the other end of the ethernet cable to the host computer.

Step 3 Set up your computer as a DHCP client, to obtain the IP address of the switch automatically. You should get an IP address within the 192.168.1.x/24 range.

It may take up to three mins. You must complete the Day 0 setup through the web UI before using the device terminal.

Step 4 Launch a web browser on the PC and enter the device IP address (`https://192.168.1.1`) in the address bar.

Step 5 Enter the Day 0 **username** and **password**.

Note
By default, the login username is *admin*, and the password is the *system serial number*. You can change it as required.

Creating User Accounts

Setting a username and password is the first task you will perform on your device. Typically, as a network administrator, you will want to control access to your device and prevent unauthorized users from seeing your network configuration or manipulating your settings.

Procedure

Step 1 Launch a web browser on the computer and enter the device IP address (https://192.168.1.1) in the address bar.

Step 2 Enter the **username** in the **Login Name** field.

Step 3 Enter **password** in the **Login User Password** field.

The username password combination gives you privilege 15 access. The string cannot start with a number, is case sensitive, and allows spaces but ignores leading spaces.

Step 4 Reconfirm the password for the user in the **Confirm Login User Password** field.

Step 5 Use the **Command Line Password** drop-down list to choose where to synchronize the password.

Step 6 In the Device ID Settings section, enter a value in the **Device Name** field.

Step 7 (Optional) Enter NTP server details in the **NTP Server** field.

Step 8 Use the **Date & Time Mode** drop-down list to select NTP time.

Choosing Setup Options

Select **Wired Network** to configure your device based on a site profile, and continue to configure switch wide settings. Otherwise, continue to the next step and configure only basic settings for your device.

Configuring Basic Device Settings

On the **Basic Device Settings** page configure the following information:

Procedure

Step 1 In the **Device Management Settings** section, assign an IP address to the management interface using either *Static* or *DHCP* address.

Step 2 If you chose *Static*, perform the following steps:

a) Enter a value in the **VLAN ID** field to associate with the interface.
b) Enter a value in the **IP Address** field. Ensure that the IP address you assign is part of the subnet mask you enter.
c) Enter a value in the **Subnet Mask** field.
d) (Optional) Enter a value in the **Default Gateway** field.
e) Use the **Associated VLAN with Interface** section to select interfaces.
f) (Optional) Use the slider next to the **Telnet** field to enable access to the device using telnet.
g) (Optional) Use the slider next to the **SSH** field to enable secure remote access to the device using Secure Shell (SSH).
h) (Optional) Use the slider next to the **SSH** field to enable secure remote access to the device using Secure Shell (SSH).
i) (Optional) Use the slider next to the **VTP Transparent Mode** field to manage VLANs across a network of switches.
j) (Optional) In the Device CIP Settings section, use the slider next to the **CIP Status** field to enable CIP.

   CIP is used for monitoring and diagnosing the health and functionality of industrial networks and devices.

Step 3 If you chose *DHCP*, perform the following steps:

a) Enter a value in the **VLAN ID** field to associate with the interface.

   VLAN ID must be a value other than 1.

b) Enter a value in the **IP Address** field to specify the default gateway. Ensure that the IP address you assign is part of the subnet mask you enter.
c) Enter a value in the **Subnet Mask** field.
d) (Optional) Enter a value in the **Default Gateway** field.
e) (Optional) Use the slider next to the **Telnet** field to enable access to the device using telnet.
f) (Optional) Use the slider next to the **SSH** field to enable secure remote access to the device using SSH.
g) (Optional) Enter a value in the **Domain Name for SSH** field.
h) (Optional) Use the slider next to the **VTP Transparent Mode** field to manage VLANs across a network of switches.
i) (Optional) In the Device CIP Settings section, use the slider next to the **CIP Status** field to enable CIP.

   CIP is used for monitoring and diagnosing the health and functionality of industrial networks and devices.

Configuring VLAN Settings

In the **VLAN Configuration** section, you can configure both data and voice VLANs.

Procedure

Step 1 Use the slider next to the **Data VLAN** to enable data VLAN.

Step 2 Use the slider next to the **Voice VLAN** to enable voice VLAN.

Configuring STP Settings

Procedure

Step 1 RPVST is the default STP mode configured on your device. You can change it to PVST from the **STP Mode** drop-down list.

Step 2 To change a bridge priority number from the default value 32768, change **Bridge Priority** to Yes and choose a priority number from the drop-down list.

Configuring DHCP, NTP, DNS and SNMP Settings

Procedure

Step 1 In the **Domain Details** section, enter a domain name that the software uses to complete unqualified hostnames.

Step 2 In the **DNS Server** field, type an IP address to identify the DNS server. This server is used for name and address resolution on your device.

Step 3 In the **DHCP Server** field, type the IP address of the DHCP server that you want to make available to DHCP clients.

Step 4 In the **Syslog Server** field, type the IP address of the server to which you want to send syslog messages.

Step 5 In the **Management Details** section, type an IP address to identify the SNMP server in the **SNMP Server** field.

SNMPv1, SNMPv2, and SNMPv3 are supported on your device.

Step 6 Specify the **SNMP community** string to permit access to the SNMP protocol.

Step 7 Click **Day 0 Config Summary** button to verify the configuration.

Step 8 Click **Submit**.

Configuring VTY Lines

For connecting to the device through Telnet or SSH, the Virtual Terminal Lines or Virtual TeleType (VTY) is used. The number of VTY lines is the maximum number of simultaneous access to the device remotely. If

the device is not configured with sufficient number of VTY lines, users might face issues with connecting to the WebUI. The default value for VTY Line is 0-15. The device allows up to 98 simultaneous sessions.

Procedure

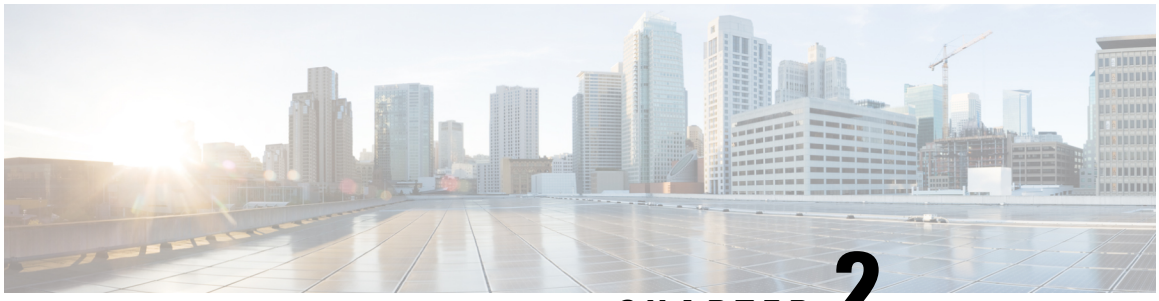
Step 1 From the WebUI, navigate through **Administration > Device > HTTP/HTTPS/Netconf/VTY**.

Step 2 In the **VTY Line** field, enter `0-xx`, depending on how many VTY lines you want to configure.

Step 3 Use the **VTY Transport Mode** drop-down list to select the VTY transport mode.

PART I

Platform



CHAPTER 2

Configuring Interface Characteristics

Information About Interface Characteristics

The following sections provide information about interface characteristics.

Interface Types

This section describes the different types of interfaces supported by the device. The rest of the chapter describes configuration procedures for physical interface characteristics.

Port-Based VLANs

A VLAN is a switched network that is logically segmented by function, team, or application, without regard to the physical location of the users. Packets received on a port are forwarded only to ports that belong to the same VLAN as the receiving port. Network devices in different VLANs cannot communicate with one another without a Layer 3 device to route traffic between the VLANs.

VLANs provide access control for traffic, and each VLAN has its own MAC address table. A VLAN comes into existence when a local port is configured to be associated with the VLAN, when the VLAN Trunking Protocol (VTP) learns of its existence from a neighbor on a trunk, or when a user creates a VLAN.

To configure VLANs, use the **vlan** *vlan-id* global configuration command to enter VLAN configuration mode. The VLAN configurations for normal-range VLANs (VLAN IDs 1 to 1005) are saved in the VLAN database. If VTP is version 1 or 2, to configure extended-range VLANs (VLAN IDs 1006 to 4094), you must first set VTP mode to transparent. Extended-range VLANs created in transparent mode are not added to the VLAN database but are saved in the switch running configuration. With VTP version 3, you can create extended-range VLANs in client or server mode in addition to transparent mode. These VLANs are saved in the VLAN database.

Add ports to a VLAN by using the **switchport** command in interface configuration mode.

- Identify the interface.
- For a trunk port, set trunk characteristics, and, if desired, define the VLANs to which it can belong.

- For an access port, set and define the VLAN to which it belongs.

Switch Ports

Switch ports are Layer 2-only interfaces associated with a physical port. Switch ports belong to one or more VLANs. A switch port can be an access port or a trunk port. You can configure a port as an access port or trunk port or let the Dynamic Trunking Protocol (DTP) operate on a per-port basis to set the switchport mode by negotiating with the port on the other end of the link. Switch ports are used for managing the physical interface and associated Layer 2 protocols and do not handle routing or bridging.

Configure switch ports by using the **switchport** interface configuration commands.

Access Ports

An access port belongs to and carries the traffic of only one VLAN (unless it is configured as a voice VLAN port). Traffic is received and sent in native formats with no VLAN tagging. Traffic arriving on an access port is assumed to belong to the VLAN assigned to the port. If an access port receives a tagged packet (Inter-Switch Link [ISL] or IEEE 802.1Q tagged), the packet is dropped, and the source address is not learned.

The types of access ports supported are:

- Static access ports are manually assigned to a VLAN (or through a RADIUS server for use with IEEE 802.1x.

You can also configure an access port with an attached Cisco IP Phone to use one VLAN for voice traffic and another VLAN for data traffic from a device attached to the phone.

Trunk Ports

A trunk port carries the traffic of multiple VLANs and by default is a member of all VLANs in the VLAN database. The IEEE 802.1Q trunk port type is supported. An IEEE 802.1Q trunk port supports simultaneous tagged and untagged traffic. An IEEE 802.1Q trunk port is assigned a default port VLAN ID (PVID), and all untagged traffic travels on the port default PVID. All untagged traffic and tagged traffic with a NULL VLAN ID are assumed to belong to the port default PVID. A packet with a VLAN ID equal to the outgoing port default PVID is sent untagged. All other traffic is sent with a VLAN tag.

Although by default, a trunk port is a member of every VLAN known to the VTP, you can limit VLAN membership by configuring an allowed list of VLANs for each trunk port. The list of allowed VLANs does not affect any other port but the associated trunk port. By default, all possible VLANs (VLAN ID 1 to 4094) are in the allowed list. A trunk port can become a member of a VLAN only if VTP knows of the VLAN and if the VLAN is in the enabled state. If VTP learns of a new, enabled VLAN and the VLAN is in the allowed list for a trunk port, the trunk port automatically becomes a member of that VLAN and traffic is forwarded to and from the trunk port for that VLAN. If VTP learns of a new, enabled VLAN that is not in the allowed list for a trunk port, the port does not become a member of the VLAN, and no traffic for the VLAN is forwarded to or from the port.

Tunnel Ports

Tunnel ports are used in IEEE 802.1Q tunneling to segregate the traffic of customers in a service-provider network from other customers who are using the same VLAN number. You configure an asymmetric link from a tunnel port on a service-provider edge switch to an IEEE 802.1Q trunk port on the customer switch. Packets entering the tunnel port on the edge switch, already IEEE 802.1Q-tagged with the customer VLANs, are encapsulated with another layer of an IEEE 802.1Q tag (called the metro tag), containing a VLAN ID unique in the service-provider network, for each customer. The double-tagged packets go through the

service-provider network keeping the original customer VLANs separate from those of other customers. At the outbound interface, also a tunnel port, the metro tag is removed, and the original VLAN numbers from the customer network are retrieved.

Tunnel ports cannot be trunk ports or access ports and must belong to a VLAN unique to each customer.

Routed Ports

A routed port is a physical port that acts like a port on a router; it does not have to be connected to a router. A routed port is not associated with a particular VLAN, as is an access port. A routed port behaves like a regular router interface, except that it does not support VLAN subinterfaces. Routed ports can be configured with a Layer 3 routing protocol. A routed port is a Layer 3 interface only and does not support Layer 2 protocols, such as DTP and STP.

Configure routed ports by putting the interface into Layer 3 mode with the **no switchport** interface configuration command. Then assign an IP address to the port, enable routing, and assign routing protocol characteristics by using the **ip routing** and **router** *protocol* global configuration commands.

Note Entering a **no switchport** interface configuration command shuts down the interface and then re-enables it, which might generate messages on the device to which the interface is connected. When you put an interface that is in Layer 2 mode into Layer 3 mode, the previous configuration information related to the affected interface might be lost.

Note A port configured as a switchport does not support MAC address configuration. It does not support the **mac-address x.x.x** command.

The number of routed ports that you can configure is not limited by software. However, the interrelationship between this number and the number of other features being configured might impact CPU performance because of hardware limitations.

Switch Virtual Interfaces

A switch virtual interface (SVI) represents a VLAN of switch ports as one interface to the routing function in the system. You can associate only one SVI with a VLAN. You configure an SVI for a VLAN only to route between VLANs or to provide IP host connectivity to the device. By default, an SVI is created for the default VLAN (VLAN 1) to permit remote device administration. Additional SVIs must be explicitly configured.

Note You cannot delete interface VLAN 1.

SVIs provide IP host connectivity only to the system. SVIs are created the first time that you enter the **vlan** interface configuration command for a VLAN interface. The VLAN corresponds to the VLAN tag associated with data frames on an ISL or IEEE 802.1Q encapsulated trunk or the VLAN ID configured for an access port. Configure a VLAN interface for each VLAN for which you want to route traffic, and assign it an IP address.

You can also use the interface range command to configure existing VLAN SVIs within the range. The commands entered under the interface range command are applied to all existing VLAN SVIs within the range. You can enter the command **interface range create vlan** *x - y* to create all VLANs in the specified

range that do not already exist. When the VLAN interface is created, **interface range vlan** *id* can be used to configure the VLAN interface.

Although the device supports a total of 1005 VLANs and SVIs, the interrelationship between the number of SVIs and routed ports and the number of other features being configured might impact CPU performance because of hardware limitations.

When you create an SVI, it does not become active until it is associated with a physical port.

EtherChannel Port Groups

EtherChannel port groups treat multiple switch ports as one switch port. These port groups act as a single logical port for high-bandwidth connections between devices or between devices and servers. An EtherChannel balances the traffic load across the links in the channel. If a link within the EtherChannel fails, traffic previously carried over the failed link changes to the remaining links. You can group multiple trunk ports into one logical trunk port, group multiple access ports into one logical access port, group multiple tunnel ports into one logical tunnel port, or group multiple routed ports into one logical routed port. Most protocols operate over either single ports or aggregated switch ports and do not recognize the physical ports within the port group. Exceptions are the DTP, the Cisco Discovery Protocol (CDP), and the Port Aggregation Protocol (PAgP), which operate only on physical ports.

When you configure an EtherChannel, you create a port-channel logical interface and assign an interface to the EtherChannel. For Layer 3 interfaces, you manually create the logical interface by using the **interface port-channel** global configuration command. Then you manually assign an interface to the EtherChannel by using the **channel-group** interface configuration command. For Layer 2 interfaces, use the **channel-group** interface configuration command to dynamically create the port-channel logical interface. This command binds the physical and logical ports together.

SKU Details

Table 1: Cisco IE3500 Series Switch SKU details

PID	Uplink Port			Downlink Port			PoE	FPGA
	Type	Ports	Interface Name	Type	Ports	Interface Name		
IE-3500-8T3S	SFP/ SFP+	3	Gigabit Ethernet 1/1-3	Copper RJ45	8	Gigabit Ethernet 1/4-11	No	No
IE-3500-8P3S							Yes	No
IE-3505-8T3S							No	Yes
IE-3505-8P3S							Yes	Yes
IE-3500-8U3X			TenGigbit Ethernet 1/1-3				Yes (4PPoE)	No
IE-3500-8T3X							No	No

PID	Uplink Port			Downlink Port			PoE	FPGA
	Type	Ports	Interface Name	Type	Ports	Interface Name		
IE-3500H-8T	Copper RJ45	3	Gigabit Ethernet 1/1-3	Copper RJ45	5	Gigabit Ethernet 1/4-8	No	No
IE-3500H-16T					13	Gigabit Ethernet 1/4-16		
IE-3505H-16T					13	Gigabit Ethernet 1/4-16		Yes
IE-3500H-24T					21	Gigabit Ethernet 1/4-24		No
IE-3500H-12FT4T		4	Gigabit Ethernet 1/1-4		12	FastEthernet 1/5-16		
IE-3500H-20FT4T					20	FastEthernet 1/5-24		
IE-3500H-14P2T		2	Gigabit Ethernet 1/1-2		14	Gigabit Ethernet 1/3-16	Yes	
IE-3500H-12P2MU2X	SFP/ SFP+	2	TenGigabit Ethernet 1/15-16		14	Gigabit Ethernet 1/1-12		
						TwoGigabit Ethernet 1/13-14	Yes (4PPoE)	

PID	Uplink Port			Downlink Port			PoE	FPGA
	Type	Ports	Interface Name	Type	Ports	Interface Name		
IEM-3500-16P	Not Applicable			Copper RJ45	16	Gigabit Ethernet 2/1-16	Yes	Yes, if connected to IE3505 SKU.
IEM-3500-16T					16	Gigabit Ethernet 2/1-16	No	
IEM-3500-8P					8	Gigabit Ethernet 2/1-8	Yes	
IEM-3500-8T					8	Gigabit Ethernet 2/1-8	No	
IEM-3500-4MU[1]					4	TwoGigabit Ethernet 2/1-4	Yes (4PPoE)	
IEM-3500-8S				SFP	8	Gigabit Ethernet 2/1-8	No	
IEM-3500-14T2S				Copper RJ45/ SFP	Copper RJ45: 14 SFP: 2	Gigabit Ethernet 2/1-16		
IEM-3500-6T2S					Copper RJ45: 6 SFP: 2	Gigabit Ethernet 2/1-8		

[1] HSR, PRP, and DLR are not supported on the IEM-3500-4MU= expansion module.

The uplink ports on the switches are fixed and correspond to SFP ports, meaning they require an SFP module, either copper or fiber, for operation.

PID decode details

Use the information below to decode the PID details.

Table 2: PID type

PID Type	Description
IE3500	Basic SKU
IE3505	Advance SKU
IE3500H	Basic SKU Heavy Duty
IE3505H	Advance SKU Heavy Duty
IEM	Expansion Module[2]

[2] HSR is not supported with expansion modules.

Table 3: Port type

Port Type	Description
T	Copper
P	PoE Copper
U	4PPoE Copper
S	SFP (Copper/Fiber)
X	SFP 10G (Copper/Fiber)
MU	Mgig 4PPoE Copper
FT	FastEthernet

PID Syntax: <PID Type>–<number of Ports> <Port Type> <number of Ports> <Port Type>

For example, see the below PID IE3505-8T3S breakdown:

- IE3505: Indicates the PID type as an advance SKU.
- 8T3S indicates the following:
  - 8: The number of downlink ports.
  - T: The downlink port type as Copper.
  - 3: The number of uplink ports.
  - S: The uplink port type as SFP (Copper/Fiber).

Power over Ethernet

The Power over Ethernet (PoE) technology allows PoE (802.3af standard), PoE+ (802.3at), and Universal Power over Ethernet Plus (UPoE+, 802.3bt) ports to supply power for the operation of a device.

For more information, see the *Configuring PoE* section of this guide.

Switch Ports

This topic provides information about using the switch ports on Cisco IE3500 Series Switch.

Console Port

The device has the following console ports:

- Cisco IE3500H Heavy Duty Series Switch
  - M12 A-coded 5-pin connector console port.
- Cisco IE3500 Rugged Series Switch
  - RJ-45 console port
  - USB mini-Type C serial console
  - USB Type A port.

Console output appears on devices connected to both ports, but console input is active on only one port at a time. By default, the USB connector takes precedence over the RJ-45 connector.

Note Windows PCs require a driver for the USB port. See the hardware installation guide for driver installation instructions.

Use the supplied USB Type A-to-USB mini-Type C cable to connect a PC or other device to the device. The connected device must include a terminal emulation application. When the device detects a valid USB connection to a powered-on device that supports host functionality (such as a PC), input from the RJ-45 console is immediately disabled, and input from the USB console is enabled. Removing the USB connection immediately reenables input from the RJ-45 console connection. An LED on the device shows which console connection is in use.

Console Port Change Logs

At software startup, a log shows whether the USB or the RJ-45 console is active. Every device always first displays the RJ-45 media type.

In the sample output, device 1 has a connected USB console cable. Because the bootloader did not change to the USB console, the first log from the device shows the RJ-45 console. A short time later, the console changes and the USB console log appears.

```
switch
*Mar  1 00:01:00.171: %USB_CONSOLE-6-MEDIA_RJ45: Console media-type is RJ45.
*Mar  1 00:01:00.431: %USB_CONSOLE-6-MEDIA_USB: Console media-type is USB.
```

When the USB cable is removed or the PC de-activates the USB connection, the hardware automatically changes to the RJ-45 console interface.

You can configure the console type to always be RJ-45, and you can configure an inactivity timeout for the USB connector.

USB Type A Port

The USB Type A port provides access to external USB flash devices, also known as thumb drives or USB keys. The port supports Cisco USB flash drives with capacities from 128 MB to 16 GB (USB devices with port densities of 128 MB, 256 MB, 1 GB, 4 GB, 8 GB, and 16 GB are supported). You can use standard Cisco IOS command- line interface (CLI) commands to read, write, erase, and copy to or from the flash device.

Disabling USB Ports

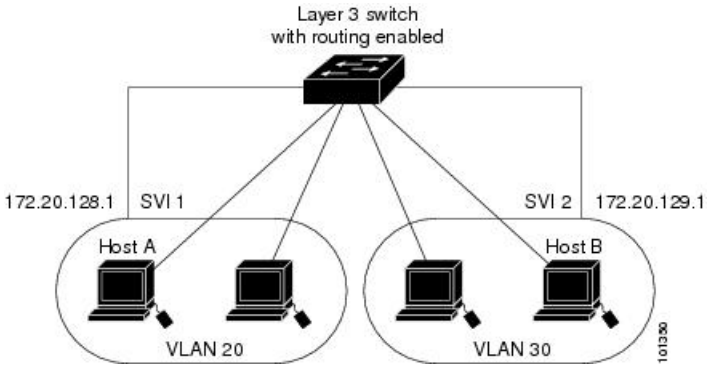
All the USB ports in a standalone device can be disabled using the **platform usb disable** command. Configuring this command disables all external media, including both USB flash and SD flash devices. To reenable the USB ports, use the **no platform usb disable** command.

When a USB port is disabled, no system messages are generated if a USB is inserted.

Interface Connections

Devices within a single VLAN can communicate directly through any switch. Ports in different VLANs cannot exchange data without going through a routing device. With a standard Layer 2 device, ports in different VLANs have to exchange information through a router. By using the device with routing enabled, when you configure both VLAN 20 and VLAN 30 with an SVI to which an IP address is assigned, packets can be sent from Host A to Host B directly through the device with no need for an external router.

Figure 1: Connecting VLANs with a Switch



VLAN routing can be achieved in two ways:

- **With a Network Advantage license**: Advanced routing protocols such as OSPF, EIGRP, and BGP are supported
- **With a Network Essentials license**: Basic routing protocols like RIP and static routing can be used.

The routing function can be enabled on all SVIs. The device routes only IP traffic. When IP routing protocol parameters and address configuration are added to an SVI, any IP traffic received from these ports is routed.

Interface Configuration Mode

The device supports these interface types:

- Physical ports: Device ports and routed ports
- VLANs: Switch virtual interfaces
- Port channels: EtherChannel interfaces

You can also configure a range of interfaces.

Note Before inserting a new SFP, you must clear any existing (speed and duplex) configuration on the port. If you fail to clear the port, any user configuration done before the insertion of the SFP may show undefined behavior.

To configure a physical interface (port), specify the interface type, module number, and device port number, and enter interface configuration mode.

- Type: Gigabit Ethernet (GigabitEthernet or gi) for 10/100/1000 Mbps Ethernet ports and small form-factor pluggable (SFP) module Gigabit Ethernet interfaces.
- On a device with SFP uplink ports, the SFP Uplink ports starts form GigabitEthernet1/1 through GigabitEthernet1/3 and Fixed Downlink ports starts from GigabitEthernet1/4 through GigabitEthernet1/11

You can identify physical interfaces by physically checking the interface location on the device. You can also use the **show** privileged EXEC commands to display information about a specific interface or all the interfaces on the switch. The remainder of this chapter primarily provides physical interface configuration procedures.

These are examples of how to configure interfaces on standalone device:

- To configure 10/100/1000 port 4 on a standalone device, enter this command:

```
Device# configure terminal
Device(config)# interface GigabitEthernet1/4
```

- To configure the first SFP module (uplink) port on a standalone device, enter this command:

```
Device# configure terminal
Device(config)# interface GigabitEthernet 1/1
```

Default Ethernet Interface Configuration

To configure Layer 2 parameters, if the interface is in Layer 3 mode, you must enter the **switchport** interface configuration command without any parameters to put the interface into Layer 2 mode. This shuts down the interface and then re-enables it, which might generate messages on the device to which the interface is connected. When you put an interface that is in Layer 3 mode into Layer 2 mode, the previous configuration information related to the affected interface might be lost, and the interface is returned to its default configuration.

This table shows the Ethernet interface default configuration, including some features that apply only to Layer 2 interfaces.

Table 4: Default Layer 2 Ethernet Interface Configuration

Feature	Default Setting
Operating mode	Layer 2 or switching mode (**switchport** command).
Allowed VLAN range	VLANs 1 to 4094.
Default VLAN (for access ports)	VLAN 1 (Layer 2 interfaces only).
Native VLAN (for IEEE 802.1Q trunks)	VLAN 1 (Layer 2 interfaces only).
VLAN trunking	Switchport mode dynamic auto (supports DTP) (Layer 2 interfaces only).
Port enable state	All ports are enabled.
Port description	None defined.
Speed	Autonegotiate.
Duplex mode	Autonegotiate.
Flow control	Flow control is not supported for sent packets.
EtherChannel (PAgP)	Disabled on all Ethernet ports.
Port blocking (unknown multicast and unknown unicast traffic)	Disabled (not blocked) (Layer 2 interfaces only).
Broadcast, multicast, and unicast storm control	Disabled.
Protected port	Disabled (Layer 2 interfaces only).
Port security	Disabled (Layer 2 interfaces only).
Port Fast	Disabled.
Auto-MDIX	Enabled.
Power over Ethernet (PoE)	Enabled (auto).

Interface Speed and Duplex Mode

Ethernet interfaces on the switch operate at 10, 100, 1000 Mbps speed and in either full-duplex or half-duplex mode. In full-duplex mode, two stations can send and receive traffic at the same time. Normally, 10-Mbps ports operate in half-duplex mode, which means that stations can either receive or send traffic.

The switch support speeds at 100 Mb and 1 Gb and operate at full-duplex mode on SFP modules that support speeds up to 1 Gbps. For the list of supported switch models, refer *Cisco IE3500 Series Switch Hardware Installation Guide*.

Speed and Duplex Configuration Guidelines

When configuring an interface speed and duplex mode, note these guidelines:

- Gigabit Ethernet (10/100/1000 Mbps) ports support all speed options and all duplex options (auto, half, and full). However, Gigabit Ethernet ports operating at 1000 Mbps and above do not support half-duplex mode.

  Multi-Gigabit Ethernet ports (2.5 Gbps) support all speed options, but only support auto and full duplex mode. These ports do not support half-duplex mode at any speed.

  SFP ports operating at 1 Gbps, SFP+ ports operating at 10 Gbps support only the **no speed nonegotiate** or **speed nonegotiate** commands. Duplex options are not supported.

Note SFP and SFP+ ports support speed (auto, 10, 100, 1000) and duplex (auto/full/half) options only if the 1000Base-T SFP is used. SFP ports support speed (auto/100) and duplex (auto/full/half) options only if the GLC-GE-100FX modules are used.

- If both ends of the line support autonegotiation, we highly recommend the default setting of **auto** negotiation.
- If one interface supports autonegotiation and the other end does not, configure duplex and speed on both interfaces; do not use the **auto** setting on the supported side.
- When STP is enabled and a port is reconfigured, the device can take up to 30 seconds to check for loops. The port LED is amber while STP reconfigures. As best practice, we suggest configuring the speed and duplex options on a link to **auto** or to **fixed** on both the ends. If one side of the link is configured to **auto** and the other side is configured to **fixed**, the link will not be up; this is expected behavior.

Caution Changing the interface speed and duplex mode configuration might shut down and re-enable the interface during the reconfiguration.

IEEE 802.3x Flow Control

Flow control enables connected Ethernet ports to control traffic rates during congestion by allowing congested nodes to pause link operation at the other end. If one port experiences congestion and cannot receive any more traffic, it notifies the other port by sending a pause frame to stop sending until the condition clears. Upon receipt of a pause frame, the sending device stops sending any data packets, which prevents any loss of data packets during the congestion period.

Note The switch ports can receive, but not send, pause frames.

You use the **flowcontrol** interface configuration command to set the interface's ability to **receive** pause frames to **on**, **off**, or **desired**. The default state is **on**.

When set to **desired**, an interface can operate with an attached device that is required to send flow-control packets or with an attached device that is not required to but can send flow-control packets.

These rules apply to flow control settings on the device:

- **receive on** (or **desired**): The port cannot send pause frames but can operate with an attached device that is required to or can send pause frames; the port can receive pause frames.
- **receive off**: Flow control does not operate in either direction. In case of congestion, no indication is given to the link partner, and no pause frames are sent or received by either device.

Note For details on the command settings and the resulting flow control resolution on local and remote ports, see the **flowcontrol** interface configuration command in the command reference for this release.

Layer 3 Interfaces

The device supports these types of Layer 3 interfaces:

- SVIs: Configure SVIs for any VLANs for which you want to route traffic. SVIs are created when you enter a VLAN ID following the **interface vlan** global configuration command. To delete an SVI, use the **no interface vlan** global configuration command.

Note

- VLAN 1 interface is the default interface and cannot be deleted.
- When you create an SVI, it does not become active until it is associated with a physical port.
- SVI MAC addresses do not change after a device reload. This is expected behavior.

When configuring SVIs, you can use the **switchport autostate exclude** command on a port to exclude that port from being included in determining SVI line-state. To disable autostate on the SVI, use the **no autostate** command on the SVI.

- Routed ports: Routed ports are physical ports configured to be in Layer 3 mode by using the **no switchport** interface configuration command.
- Layer 3 EtherChannel ports: EtherChannel interfaces made up of routed ports.

A Layer 3 device can have an IP address assigned to each routed port and SVI.

There is no defined limit to the number of SVIs and routed ports that can be configured in a device. However, the interrelationship between the number of SVIs and routed ports and the number of other features being configured might have an impact on CPU usage because of hardware limitations. If the device is using its maximum hardware resources, attempts to create a routed port or SVI have these results:

- If you try to create a new routed port, the device generates a message that there are not enough resources to convert the interface to a routed port, and the interface remains as a switchport.
- If you try to create an extended-range VLAN, an error message is generated, and the extended-range VLAN is rejected.

- If the device is notified by VLAN Trunking Protocol (VTP) of a new VLAN, it sends a message that there are not enough hardware resources available and shuts down the VLAN. The output of the **show vlan** user EXEC command shows the VLAN in a suspended state.
- If the device attempts to boot up with a configuration that has more VLANs and routed ports than hardware can support, the VLANs are created, but the routed ports are shut down, and the device sends a message that this was due to insufficient hardware resources.

Note All Layer 3 interfaces require an IP address to route traffic. This procedure shows how to configure an interface as a Layer 3 interface and how to assign an IP address to an interface:

If the physical port is in Layer 2 mode (the default), you must enter the **no switchport** interface configuration command to put the interface into Layer 3 mode. Entering a **no switchport** command disables and then re-enables the interface, which might generate messages on the device to which the interface is connected. Furthermore, when you put an interface that is in Layer 2 mode into Layer 3 mode, the previous configuration information related to the affected interface might be lost, and the interface is returned to its default configuration.

How to Configure Interface Characteristics

The following sections provide information about the various tasks that comprise the procedure to configure interface characteristics.

Configuring an Interface

These general instructions apply to all interface configuration processes.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** **Example:** `Device(config)# interface gigabitethernet1/1` `Device(config-if)#`	Identifies the interface type, and the number of the connector.

	Command or Action	Purpose
Step 4	Follow each **interface** command with the interface configuration commands that the interface requires.	Defines the protocols and applications that will run on the interface. The commands are collected and applied to the interface when you enter another interface command or enter **end** to return to privileged EXEC mode.
Step 5	**show interfaces**	Displays a list of all interfaces on or configured for the switch. A report is provided for each interface that the device supports or for the specified interface.

Adding a Description for an Interface

Follow these steps to add a description for an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface for which you are adding a description, and enter interface configuration mode.
Step 4	**description** *string* **Example:** `Device(config-if)# description Connects to Marketing`	Adds a description for an interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show interfaces** *interface-id* **description**	Verifies your entry.

	Command or Action	Purpose
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Range of Interfaces

To configure multiple interfaces with the same configuration parameters, use the **interface range** global configuration command. When you enter the interface-range configuration mode, all command parameters that you enter are attributed to all interfaces within that range until you exit this mode.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface range** {**GigabitEthernet** | **Loopback** | **Port-channel** | **Tunnel** | **create** | *macro*}} **Example:** `Device(config)# interface range GigabitEthernet 1/1-11, GigabitEthernet 2/1-8`	Specifies the range of interfaces (VLANs or physical ports) to be configured, and enter interface-range configuration mode. For example, an 8 port expansion module can use: *GigabitEthernet 2/1-8*. • You can use the **interface range** command to configure up to five port ranges or a previously defined macro. • The **macro** variable is explained in Configuring and Using Interface Range Macros. • In a comma-separated *port-range*, you must enter the interface type for each entry and enter spaces before and after the comma. • In a hyphen-separated *port-range*, you do not need to re-enter the interface type, but you must enter a space before the hyphen.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring and Using Interface Range Macros

You can create an interface range macro to automatically select a range of interfaces for configuration. Before you can use the **macro** keyword in the **interface range macro** global configuration command string, you must use the **define interface-range** global configuration command to define the macro.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**define interface-range** *macro_name interface-range* **Example:** `Device(config)# define interface-range enet_list gigabitethernet1/1-11`	Defines the interface-range macro, and saves it in NVRAM. • The *macro_name* is a 32-character maximum character string. • A macro can contain up to five comma-separated interface ranges. • Each *interface-range* must consist of the same port type. **Note** Before you can use the **macro** keyword in the **interface range macro** global configuration command string, you must use the **define interface-range** global configuration command to define the macro.

	Command or Action	Purpose
Step 4	**interface range macro** *macro_name* **Example:** `Device(config)# interface range macro enet_list`	Selects the interface range to be configured using the values saved in the interface-range macro called *macro_name*. You can now use the normal configuration commands to apply the configuration to all interfaces in the defined macro.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config | include define** **Example:** `Device# show running-config | include define`	Shows the defined interface range macro configuration.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Setting the Interface Speed and Duplex Parameters

Follow these steps to configure the interface speed and duplex parameters.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:**	Specifies the physical interface to be configured, and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet1/3`	
Step 4	**duplex** {**auto** | **full** | **half**} **Example:** `Device(config-if)# duplex half`	Enters the duplex parameter for the interface. Enables half-duplex mode (for interfaces operating only at 10 or 100 Mb/s). Half duplex is not supported on multi-Gigabit Ethernet ports configured for speed of 1000 Mb/s. You can configure the duplex setting when the speed is set to **auto**.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show interfaces** *interface-id* **Example:** `Device# show interfaces gigabitethernet1/1`	Displays the interface speed and duplex mode configuration.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring the IEEE 802.3x Flow Control

Follow these steps to configure the IEEE 802.3x flow control.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode

	Command or Action	Purpose
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/4`	Specifies the physical interface to be configured, and enters interface configuration mode.
Step 4	**flowcontrol** {**receive**} {**on** | **off** | **desired**} **Example:** `Device(config-if)# flowcontrol receive on`	Configures the flow control mode for the port.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show flowcontrol interface** *interface-id* **Example:** `Device# show flowcontrol interface GigabitEthernet1/4`	Verifies the specified interface flow control settings.
Step 7	**show flowcontrol module** *slot* **Example:** `Device# show flowcontrol module 1`	Verifies the interface flow control settings on the module.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configure Layer 3 Interface

Follow these steps to configure a layer 3 interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** {**gigabitethernet** *interface-id*} | {**vlan** *vlan-id*} | {**port-channel** *port-channel-number*} **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface to be configured as a Layer 3 interface, and enters interface configuration mode.
Step 4	**no switchport** **Example:** `Device(config-if)# no switchport`	(For physical ports only) Enters Layer 3 mode.
Step 5	**ip address** *ip_address subnet_mask* **Example:** `Device(config-if)# ip address 192.0.2.10 255.255.255.0`	Configures the IP address and IP subnet.
Step 6	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Enables the interface.
Step 7	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 8	**show interfaces** [*interface-id*]	Verifies the configuration.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Logical Layer 3 GRE Tunnel Interface

Before you begin

Generic Routing Encapsulation (GRE) is a tunneling protocol used to encapsulate network layer protocols inside virtual point-to-point links. A GRE tunnel only provides encapsulation and not encryption.

Note

- GRE tunnels are supported on the hardware. When GRE is configured without tunnel options, packets are hardware-switched. When GRE is configured with tunnel options (such as key, checksum, and so on), packets are switched in the software.
- A maximum of 10 GRE tunnels are supported.
- Other features such as Access Control Lists (ACL) and Quality of Service (QoS) are not supported for the GRE tunnels.
- The **tunnel path-mtu-discovery** command is not supported for GRE tunnels. To avoid fragmentation, you can set the maximum transmission unit (MTU) of both ends of the GRE tunnel to the lowest value by using the **ip mtu 832** command.

To configure a GRE tunnel, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *number* **Example:** `Device(config)# interface tunnel 2`	Enables tunneling on the interface.
Step 4	**ip address** *ip_address subnet_mask* **Example:** `Device(config)# ip address 192.0.2.10 255.255.255.0`	Configures the IP address and IP subnet.
Step 5	**tunnel source** {*ip_address* | *type_number*} **Example:** `Device(config)# tunnel source 10.0.0.1`	Configures the tunnel source.
Step 6	**tunnel destination** {*host_name* | *ip_address*} **Example:** `Device(config)# tunnel destination 10.0.0.2`	Configures the tunnel destination.

	Command or Action	Purpose
Step 7	**tunnel mode gre ip** **Example:** `Device(config)# tunnel mode gre ip`	Configures the tunnel mode.
Step 8	**end** **Example:** `Device(config)# end`	Exits configuration mode.

Configuring SVI Autostate Exclude

SVI Autostate controls the status of a Switch Virtual Interface (SVI), determining whether it is up or down based on the operational state of its associated switch ports. In Layer 3 switches, the **switchport autostate exclude** command automatically brings up a VLAN when configured. By default, SVI autostate is enabled. It can also be disabled to manually bring up the VLAN.

Follow these steps to exclude SVI autostate.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies a Layer 2 interface (physical port or port channel), and enters interface configuration mode.
Step 4	**switchport autostate exclude** **Example:** `Device(config-if)# switchport autostate exclude`	Excludes the access or trunk port when defining the status of an SVI line state (up or down)
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	
Step 6	**show running config interface** *interface-id* **Example:** `Device# show running config interface gigabitethernet1/1`	(Optional) Shows the running configuration. Verifies the configuration.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Shutting Down and Restarting an Interface

Shutting down an interface disables all functions on the specified interface and marks the interface as unavailable on all monitoring command displays. This information is communicated to other network servers through all dynamic routing protocols. The interface is not mentioned in any routing updates.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** {**vlan** *vlan-id*} | { **gigabitethernet** *interface-id*} | {**port-channel** *port-channel-number*} **Example:** `Device(config)# interface gigabitethernet1/1`	Selects the interface to be configured.
Step 4	**shutdown** **Example:** `Device(config-if)# shutdown`	Shuts down an interface.

	Command or Action	Purpose
Step 5	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Restarts an interface.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.

Configuring the Console Media Type

Follow these steps to set the console media type to RJ-45. If you configure the console as RJ-45, USB console operation is disabled, and input comes only through the RJ-45 connector.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**line console 0** **Example:** `Device(config)# line console 0`	Configures the console and enters line configuration mode.
Step 4	**media-type rj45** **Example:** `Device(config-line)# media-type rj45`	Configures the console media type to be only RJ-45 port. If you do not enter this command and both types are connected, the USB port is used by default.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring USB Inactivity Timeout

When the USB console port is deactivated due to a timeout, you can restore its operation by disconnecting and reconnecting the USB cable.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**line console 0** **Example:** `Device(config)# line console 0`	Configures the console and enters line configuration mode.
Step 4	**usb-inactivity-timeout** *timeout-minutes* **Example:** `Device(config-line)# usb-inactivity-timeout 30`	Specifies an inactivity timeout for the console port. The range is 1 to 240 minutes. The default is to have no timeout configured.
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Disabling USB Ports

To disable all USB ports, peform this procedure.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	[**no**] **platform usb disable** **Example:** `Device(config)# platform usb disable`	Disables all the USB ports on the device. Use the **no platform usb disable** command to reenable the USB ports.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits to privileged EXEC mode.
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring Interface Characteristics

The following sections provide information about monitoring interface characteristics.

Monitoring Interface Status

Commands entered at the privileged EXEC prompt display information about the interface, including the versions of the software and the hardware, the configuration, and statistics about the interfaces.

Table 5: show Commands for Interfaces

Command	Purpose
show interfaces *interface-id* **status** [**err-disabled**]	Displays interface status or a list of interfaces in the error-disabled state.

Command	Purpose
show interfaces [*interface-id*] **switchport**	Displays administrative and operational status of switching (nonrouting) ports. You can use this command to find out if a port is in routing or in switching mode.
show interfaces [*interface-id*] **description**	Displays the description configured on an interface or all interfaces and the interface status.
show ip interface [*interface-id*]	Displays the usability status of all interfaces configured for IP routing or the specified interface.
show interface [*interface-id*] **stats**	Displays the input and output packets by the switching path for the interface.
show interface [*interface-id*] **link**[**module** *number*]	Displays the up time and down time of an interface or all interfaces.
show interfaces *interface-id*	(Optional) Displays speed and duplex on the interface.
show interfaces transceiver dom-supported-list	(Optional) Displays Digital Optical Monitoring (DOM) status on the connect SFP modules.
show interfaces transceiver properties	(Optional) Displays temperature, voltage, or amount of current on the interface.
show interfaces [*interface-id*] [{**transceiver properties** | **detail**}] *module number*]	Displays physical and operational status about an SFP module.
show running-config interface [*interface-id*]	Displays the running configuration in RAM for the interface.
show version	Displays the hardware configuration, software version, the names and sources of configuration files, and the boot images.
show controllers ethernet-controller *interface-id* **phy**	Displays the operational state of the auto-MDIX feature on the interface.

Clearing and Resetting Interfaces and Counters

Table 6: clear Commands for Interfaces

Command	Purpose
clear counters [*interface-id*]	Clears interface counters.
clear interface *interface-id*	Resets the hardware logic on an interface.
clear line [*number* | **console 0** | **vty** *number*]	Resets the hardware logic on an asynchronous serial line.

Note The **clear counters** privileged EXEC command does not clear counters retrieved by using Simple Network Management Protocol (SNMP), but only those seen with the **show interface** privileged EXEC command.

Configuration Examples for Interface Characteristics

The following sections provide examples of interface characteristics configurations.

Example: Adding a Description to an Interface

The following example shows how to add a description to an interface:

```
Device# configure terminal
Enter configuration commands, one per line. End with CNTRL/Z.
Device(config)# interface gigabitethernet1/1
Device(config-if)# description Connects to Marketing
Device(config-if)# end
Device# show interfaces gigabitethernet1/1 description

Interface Status         Protocol Description
Gi1/1     admin down     down     Connects to Marketing
```

Example: Setting Interface Speed and Duplex Mode

The following example shows how to set the interface speed to 10 Mbps and the duplex mode to full, on a 10/100/1000 Mbps port:

```
Device# configure terminal
Device(config)# interface gigabitethernet1/4
Device(config-if)# speed 10
Device(config-if)# duplex full
```

The following example shows how to set the interface speed to 100 Mbps on a 10/100/1000 Mbps port:

```
Device# configure terminal
Device(config)# interface gigabitethernet1/4
Device(config-if)# speed 100
```

Example: Configuring a Layer 3 Interface

The following example shows how to configure a Layer 3 interface:

```
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# interface gigabitethernet1/1
Device(config-if)# no switchport
Device(config-if)# ip address 192.0.2.10 255.255.255.0
Device(config-if)# no shutdown
```

Example: Configuring the Console Media Type

The following example shows how to disable the USB console media type and enable the RJ-45 console media type:

```
Device# configure terminal
Device(config)# line console 0
Device(config-line)# media-type rj45
```

The following example shows how to reverse the previous configuration and immediately activate any USB console that is connected:

```
Device# configure terminal
Device(config)# line console 0
Device(config-line)# no media-type rj45
```

Example: Configuring USB Inactivity Timeout

The following example shows how to configure the inactivity timeout to 30 minutes:

```
Device# configure terminal
Device(config)# line console 0
Device(config-line)# usb-inactivity-timeout 30
```

```
Device# configure terminal
Device(config)# line console 0
Device(config-line)# no usb-inactivity-timeout
```

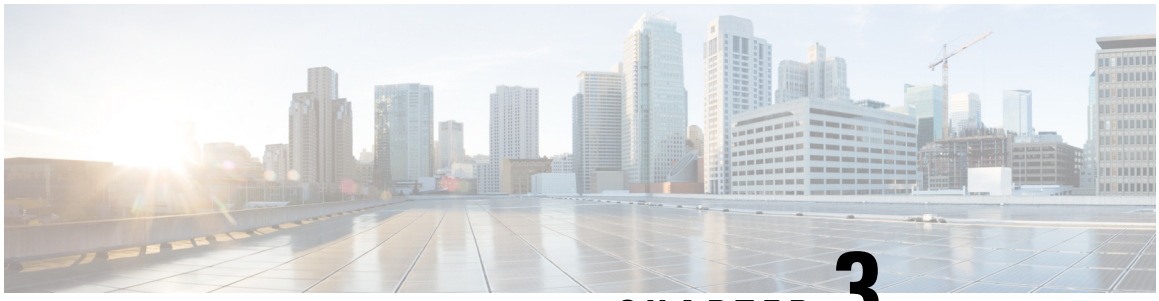
If there is no (input) activity on a USB console port for the configured number of minutes, the inactivity timeout setting applies to the RJ-45 port, and a log shows this occurrence:

```
*Mar  1 00:47:25.625: %USB_CONSOLE-6-INACTIVITY_DISABLE: Console media-type USB disabled
due to inactivity, media-type reverted to RJ45.
```

At this point, the only way to reactivate the USB console port is to disconnect and reconnect the cable.

When the USB cable on a switch is disconnected and reconnected, a log, which is similar to this, appears:

```
*Mar  1 00:48:28.640: %USB_CONSOLE-6-MEDIA_USB: Console media-type is USB.
```



CHAPTER 3

Auto-MDIX

Prerequisites for Auto-MDIX

To configure Layer 2 parameters, if the interface is in Layer 3 mode, you must enter the **switchport** interface configuration command without any parameters to put the interface into Layer 2 mode. This shuts down the interface and then re-enables it, which might generate messages on the device to which the interface is connected. When you put an interface that is in Layer 3 mode into Layer 2 mode, the previous configuration information related to the affected interface might be lost, and the interface is returned to its default configuration.

Automatic medium-dependent interface crossover (auto-MDIX) is enabled by default.

Restrictions for Auto-MDIX

- The device might not support a pre-standard powered device—such as Cisco IP phones and access points that do not fully support IEEE 802.3af—if that powered device is connected to the device through a crossover cable. This is regardless of whether auto-MIDX is enabled on the switch port.
- After each device reload, interfaces configured with the **no mdix auto** command will be in down state. To enable the interface, each time after a reload, you have to remove the SFP and reinsert the SFP.

Information About Auto-MDIX

The following sections provide information about Auto-MDIX.

Auto-MDIX on an Interface

When automatic medium-dependent interface crossover (auto-MDIX) is enabled on an interface, the interface automatically detects the required cable connection type (straight through or crossover) and configures the connection appropriately. When connecting devices without the auto-MDIX feature, you must use

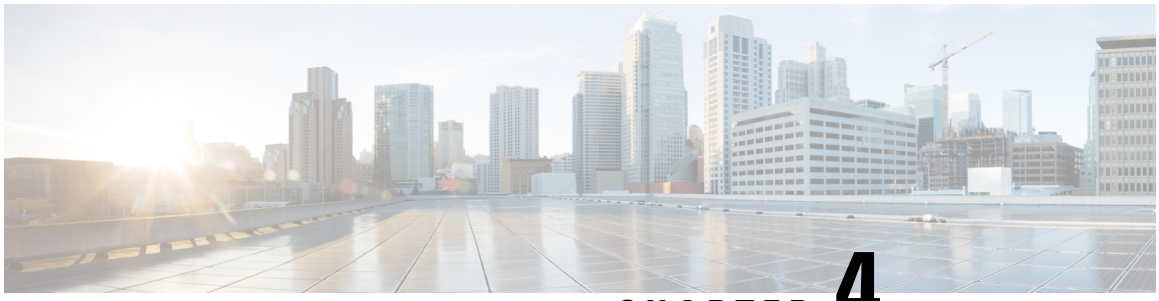
straight-through cables to connect to devices such as servers, workstations, or routers and crossover cables to connect to other devices or repeaters. With auto-MDIX enabled, you can use either type of cable to connect to other devices, and the interface automatically corrects for any incorrect cabling. For more information about cabling requirements, see the hardware installation guide.

Note Auto-MDIX is enabled by default.

This table shows the link states that result from auto-MDIX settings and correct and incorrect cabling.

Table 7: Link Conditions and Auto-MDIX Settings

Local Side Auto-MDIX	Remote Side Auto-MDIX	With Correct Cabling	With Incorrect Cabling
On	On	Link up	Link up
On	Off	Link up	Link up
Off	On	Link up	Link up
Off	Off	Link up	Link down



CHAPTER 4

Checking Port Status and Connectivity

Check Cable Status Using Time Domain Reflectometer

The Time Domain Reflectometer (TDR) feature allows you to determine if a cable is OPEN or SHORT when it is at fault.

Running the TDR Test

To start the TDR test, perform this task:

Procedure

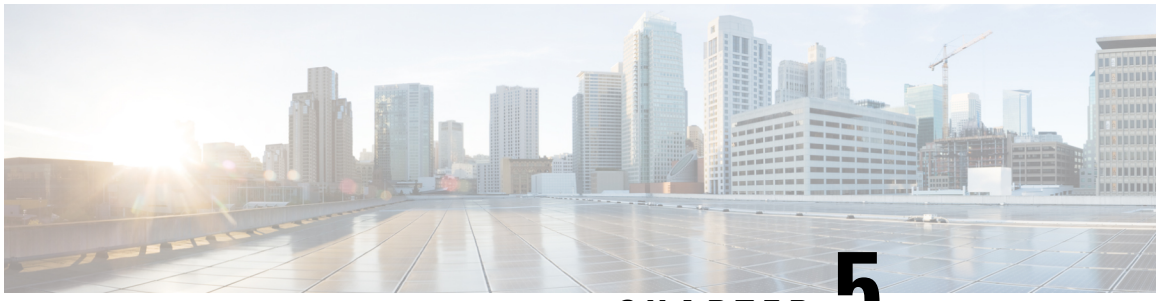
	Command or Action	Purpose
Step 1	**test cable-diagnostics tdr** {**interface** { *interface-number* }}	Starts the TDR test.
Step 2	**show cable-diagnostics tdr** {**interface** *interface-number*}	Displays the TDR test counter information.

TDR Guidelines

The following guidelines apply to the use of TDR:

- Do not change the port configuration while the TDR test is running.
- If you connect a port undergoing a TDR test to an Auto-MDIX enabled port, the TDR result might be invalid.
- If you connect a port undergoing a TDR test to a 100BASE-T port such as that on the device, the unused pairs (4-5 and 7-8) are reported as faulty because the remote end does not terminate these pairs.
- To run a TDR test, the cable length should be at least 10 meters. If the cable is shorter than 10 meters, the test is considered as invalid.
- Due to cable characteristics, you should run the TDR test multiple times to get accurate results.

- Do not change port status (for example, remove the cable at the near or far end) because the results might be inaccurate.
- TDR works best if the test cable is disconnected from the remote port. Otherwise, it might be difficult for you to interpret results correctly.
- TDR operates across four wires. Depending on the cable conditions, the status might show that one pair is OPEN or SHORT while all other wire pairs display as faulty. This operation is acceptable because you should declare a cable faulty provided one pair of wires is either OPEN or SHORT.
- TDR intent is to determine how poorly a cable is functioning rather than to locate a faulty cable.
- When TDR locates a faulty cable, you should still use an offline cable diagnosis tool to better diagnose the problem.



CHAPTER 5

Configuring LLDP and LLDP-MED

Restrictions for LLDP

- If the interface is configured as a tunnel port, LLDP is automatically disabled.
- If you first configure a network-policy profile on an interface, you cannot apply the **switchport voice vlan** command on the interface. If the **switchport voice vlan** *vlan-id* is already configured on an interface, you can apply a network-policy profile on the interface. This way the interface has the voice or voice-signaling VLAN network-policy profile applied on the interface.
- You cannot configure static secure MAC addresses on an interface that has a network-policy profile.
- When Cisco Discovery Protocol and LLDP are both in use within the same switch, it is necessary to disable LLDP on interfaces where Cisco Discovery Protocol is in use for power negotiation. LLDP can be disabled at interface level with the commands **no lldp tlv-select power-management** or **no lldp transmit / no lldp receive**.

Information About LLDP and LLDP-MED

This section describes about LLDP and LLDP-MED.

LLDP

The Cisco Discovery Protocol (CDP) is a device discovery protocol that runs over Layer 2 (the data link layer) on all Cisco-manufactured devices (routers, bridges, access servers, switches, and controllers). CDP allows network management applications to automatically discover and learn about other Cisco devices connected to the network.

To support non-Cisco devices and to allow for interoperability between other devices, the device supports the IEEE 802.1AB Link Layer Discovery Protocol (LLDP). LLDP is a neighbor discovery protocol that is used

for network devices to advertise information about themselves to other devices on the network. This protocol runs over the data-link layer, which allows two systems running different network layer protocols to learn about each other.

LLDP Supported TLVs

LLDP supports a set of attributes that it uses to discover neighbor devices. These attributes contain type, length, and value descriptions and are referred to as TLVs. LLDP supported devices can use TLVs to receive and send information to their neighbors. This protocol can advertise details such as configuration information, device capabilities, and device identity.

The switch supports these basic management TLVs. These are mandatory LLDP TLVs.

- Port description TLV
- System name TLV
- System description TLV
- System capabilities TLV
- Management address TLV

These organizationally specific LLDP TLVs are also advertised to support LLDP-MED.

- Port VLAN ID TLV (IEEE 802.1 organizationally specific TLVs)
- MAC/PHY configuration/status TLV (IEEE 802.3 organizationally specific TLVs)

LLDP-MED

LLDP for Media Endpoint Devices (LLDP-MED) is an extension to LLDP that operates between endpoint devices such as IP phones and network devices. It specifically provides support for voice over IP (VoIP) applications and provides additional TLVs for capabilities discovery, network policy, Power over Ethernet, inventory management and location information. By default, all LLDP-MED TLVs are enabled.

LLDP-MED Supported TLVs

LLDP-MED supports these TLVs:

- LLDP-MED capabilities TLV

  Allows LLDP-MED endpoints to determine the capabilities that the connected device supports and has enabled.

- Network policy TLV

  Allows both network connectivity devices and endpoints to advertise VLAN configurations and associated Layer 2 and Layer 3 attributes for the specific application on that port. For example, the switch can notify a phone of the VLAN number that it should use. The phone can connect to any device, obtain its VLAN number, and then start communicating with the call control.

  By defining a network-policy profile TLV, you can create a profile for voice and voice-signaling by specifying the values for VLAN, class of service (CoS), differentiated services code point (DSCP), and tagging mode. These profile attributes are then maintained centrally on the switch and propagated to the phone.

- Power management TLV

  Enables advanced power management between LLDP-MED endpoint and network connectivity devices. Allows devices and phones to convey power information, such as how the device is powered, power priority, and how much power the device needs.

  LLDP-MED also supports an extended power TLV to advertise fine-grained power requirements, end-point power priority, and end-point and network connectivity-device power status. LLDP is enabled and power is applied to a port, the power TLV determines the actual power requirement of the endpoint device so that the system power budget can be adjusted accordingly. The device processes the requests and either grants or denies power based on the current power budget. If the request is granted, the switch updates the power budget. If the request is denied, the device turns off power to the port, generates a syslog message, and updates the power budget. If LLDP-MED is disabled or if the endpoint does not support the LLDP-MED power TLV, the initial allocation value is used throughout the duration of the connection.

  You can change power settings by entering the **power inline** {**auto** [**max** *max-wattage*] | **never** | **static** [**max** *max-wattage*] } interface configuration command. By default the PoE interface is in **auto** mode;

- Inventory management TLV

  Allows an endpoint to send detailed inventory information about itself to the device, including information hardware revision, firmware version, software version, serial number, manufacturer name, model name, and asset ID TLV.

- Location TLV

  Provides location information from the device to the endpoint device. The location TLV can send this information:

  - Civic location information

    Provides the civic address information and postal information. Examples of civic location information are street address, road name, and postal community name information.

  - ELIN location information

    Provides the location information of a caller. The location is determined by the Emergency location identifier number (ELIN), which is a phone number that routes an emergency call to the local public safety answering point (PSAP) and which the PSAP can use to call back the emergency caller.

  - Geographic location information

    Provides the geographical details of a switch location such as latitude, longitude, and altitude of a switch.

  - custom location

    Provides customized name and value of a switch location.

Default LLDP Configuration

Table 8: Default LLDP Configuration

Feature	Default Setting
LLDP global state	Enabled

Feature	Default Setting
LLDP holdtime (before discarding)	120 seconds
LLDP timer (packet update frequency)	30 seconds
LLDP reinitialization delay	2 seconds
LLDP tlv-select	Enabled
LLDP interface state	Enabled
LLDP receive	Enabled
LLDP transmit	Enabled
LLDP med-tlv-select	Enabled. When LLDP is globally enabled, LLDP-MED-TLV is also enabled.

How to Configure LLDP and LLDP-MED

This section provides the procedures to configure LLDP and LLDP-MED.

Enabling LLDP

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**lldp run** **Example:** `Device(config)# lldp run`	Enables LLDP globally on the device.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface on which you are enabling LLDP, and enters interface configuration mode.

	Command or Action	Purpose
Step 5	**lldp transmit** **Example:** `Device(config-if)# lldp transmit`	Enables the interface to send LLDP packets.
Step 6	**lldp receive** **Example:** `Device(config-if)# lldp receive`	Enables the interface to receive LLDP packets.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode, and returns to privileged EXEC mode.
Step 8	**show lldp** **Example:** `Device# show lldp`	Verifies the configuration.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring LLDP Characteristics

You can configure the frequency of LLDP updates, the amount of time to hold the information before discarding it, and the initialization delay time. You can also select the LLDP and LLDP-MED TLVs to send and receive.

Note Steps 3 through 6 are optional and can be performed in any order.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**lldp holdtime** *seconds* **Example:** `Device(config)# lldp holdtime 120`	(Optional) Specifies the amount of time a receiving device should hold the information from your device before discarding it. The range is 0 to 65535 seconds; the default is 120 seconds.
Step 4	**lldp reinit** *delay* **Example:** `Device(config)# lldp reinit 2`	(Optional) Specifies the delay time in seconds for LLDP to initialize on an interface. The range is 2 to 5 seconds; the default is 2 seconds.
Step 5	**lldp timer** *rate* **Example:** `Device(config)# lldp timer 30`	(Optional) Sets the sending frequency of LLDP updates in seconds. The range is 5 to 65534 seconds; the default is 30 seconds.
Step 6	**lldp tlv-select** **Example:** `Device(config)# tlv-select`	(Optional) Specifies the LLDP TLVs to send or receive.
Step 7	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface on which you are enabling LLDP, and enters interface configuration mode.
Step 8	**lldp med-tlv-select** **Example:** `Device(config-if)# lldp med-tlv-select inventory management`	(Optional) Specifies the LLDP-MED TLVs to send or receive.
Step 9	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode, and returns to privileged EXEC mode.
Step 10	**show lldp** **Example:** `Device# show lldp`	Verifies the configuration.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring LLDP-MED TLVs

By default, the device only sends LLDP packets until it receives LLDP-MED packets from the end device. It then sends LLDP packets with MED TLVs, as well. When the LLDP-MED entry has been aged out, it again only sends LLDP packets.

By using the **lldp** interface configuration command, you can configure the interface not to send the TLVs listed in the following table.

Table 9: LLDP-MED TLVs

LLDP-MED TLV	Description
inventory-management	LLDP-MED inventory management TLV
location	LLDP-MED location TLV
network-policy	LLDP-MED network policy TLV
power-management	LLDP-MED power management TLV

Follow these steps to enable a TLV on an interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface on which you are enabling LLDP, and enters interface configuration mode.
Step 4	**lldp med-tlv-select** **Example:** `Device(config-if)# lldp med-tlv-select inventory management`	Specifies the TLV to enable.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits global configuration mode, and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Network-Policy TLV

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**network-policy profile** *profile number* **Example:** `Device(config)# network-policy profile 1`	Specifies the network-policy profile number, and enters network-policy configuration mode. The range is 1 to 4294967295.
Step 4	{**voice** | **voice-signaling**} **vlan** [*vlan-id* {**cos** *cvalue* | **dscp** *dvalue*}] | [[**dot1p** {**cos** *cvalue* | **dscp** *dvalue*}] | **none** | **untagged**] **Example:** `Device(config-network-policy)# voice vlan 100 cos 4`	Configures the policy attributes: • **voice**: Specifies the voice application type. • **voice-signaling**: Specifies the voice-signaling application type. • **vlan**: Specifies the native VLAN for voice traffic. • *vlan-id*: (Optional) Specifies the VLAN for voice traffic. The range is 1 to 4094. • **cos** *cvalue*: (Optional) Specifies the Layer 2 priority class of service (CoS) for the configured VLAN. The range is 0 to 7; the default is 5. • **dscp** *dvalue*: (Optional) Specifies the differentiated services code point (DSCP) value for the configured VLAN. The range is 0 to 63; the default is 46.

	Command or Action	Purpose
		• **dot1p**: (Optional) Configures the telephone to use IEEE 802.1p priority tagging and use VLAN 0 (the native VLAN). • **none**: (Optional) Do not instruct the IP telephone about the voice VLAN. The telephone uses the configuration from the telephone key pad. • **untagged**: (Optional) Configures the telephone to send untagged voice traffic. This is the default for the telephone.
Step 5	**exit** **Example:** `Device(config)# exit`	Returns to global configuration mode.
Step 6	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface on which you are configuring a network-policy profile, and enters interface configuration mode.
Step 7	**network-policy** *profile number* **Example:** `Device(config-if)# network-policy 1`	Specifies the network-policy profile number.
Step 8	**lldp med-tlv-select network-policy** **Example:** `Device(config-if)# lldp med-tlv-select network-policy`	Specifies the network-policy TLV.
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 10	**show network-policy profile** **Example:** `Device# show network-policy profile`	Verifies the configuration.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuration Examples for LLDP and LLDP-MED

This section provides configuration examples for LLDP and LLDP-MED.

Examples: Configuring Network-Policy TLV

This example shows how to configure VLAN 100 for voice application with CoS and to enable the network-policy profile and network-policy TLV on an interface:

```
Device# configure terminal
Device(config)# network-policy 1
Device(config-network-policy)# voice vlan 100 cos 4
Device(config-network-policy)# exit
Device(config)# interface gigabitethernet1/1
Device(config-if)# network-policy profile 1
Device(config-if)# lldp med-tlv-select network-policy
```

This example shows how to configure the voice application type for the native VLAN with priority tagging:

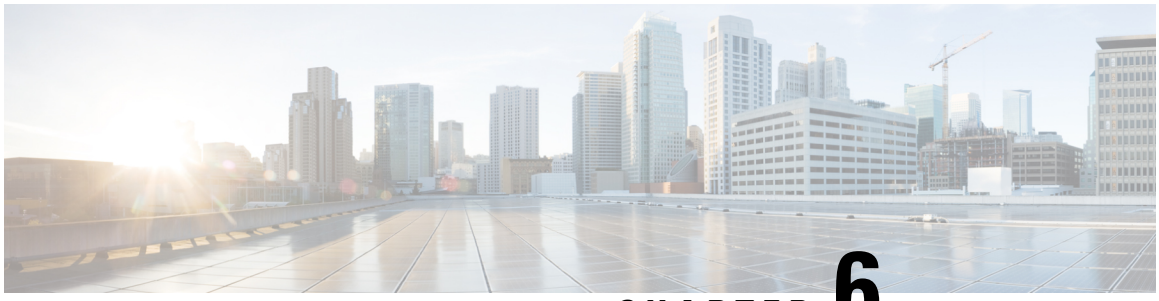
```
Device-config-network-policy)# voice vlan dot1p cos 4
Device-config-network-policy)# voice vlan dot1p dscp 34
```

Monitoring and Maintaining LLDP and LLDP-MED

Use the following commands for monitoring and maintaining LLDP and LLDP-MED.

Command	Description
clear lldp counters	Resets the traffic counters to zero.
clear lldp table	Deletes the LLDP neighbor information table.
clear nmsp statistics	Clears the NMSP statistic counters.
show lldp	Displays global information, such as frequency of transmissions, the holdtime for packets being sent, and the delay time before LLDP initializes on an interface.
show lldp entry *entry-name*	Displays information about a specific neighbor. You can enter an asterisk (*) to display all neighbors, or you can enter the neighbor name.
show lldp interface [*interface-id*]	Displays information about interfaces with LLDP enabled. You can limit the display to a specific interface.

Command	Description
show lldp neighbors [*interface-id*] [**detail**]	Displays information about neighbors, including device type, interface type and number, holdtime settings, capabilities, and port ID. You can limit the display to neighbors of a specific interface or expand the display for more detailed information.
show lldp traffic	Displays LLDP counters, including the number of packets sent and received, number of packets discarded, and number of unrecognized TLVs.
show location admin-tag *string*	Displays the location information for the specified administrative tag or site.
show location civic-location identifier *id*	Displays the location information for a specific global civic location.
show location elin-location identifier *id*	Displays the location information for an emergency location
show network-policy profile	Displays the configured network-policy profiles.



CHAPTER 6

Configuring System MTU

Information About the MTU

The default maximum transmission unit (MTU) size for payload received in Ethernet frame and sent on all device interfaces is 1500 bytes. The maximum value of System MTU is 9198 bytes.

System MTU Value Application

The upper limit of the IP or IPv6 MTU value is based on the switch configuration and refers to the currently applied system MTU value. For more information about setting the MTU sizes, see the **system mtu** global configuration command in the command reference for this release.

The minimum IPv6 system MTU is fixed at 1280 as per RFC 8200.

How to Configure MTU

The following tasks describe how you can configure MTU.

Configuring the System MTU

Follow these steps to change the MTU size for switched packets:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**system mtu** *bytes* **Example:** `Device(config)# system mtu 1900`	(Optional) Changes the MTU size for all interfaces.
Step 4	**end** **Example:** `Device(config)# end`	Enters global configuration mode, and returns to privileged EXEC mode.
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves your entries in the configuration file.
Step 6	**show system mtu** **Example:** `Device# show system mtu`	Verifies your settings.

Configuring Protocol-Specific MTU

To override system MTU values on routed interfaces, configure protocol-specific MTU under each routed interface. To change the MTU size for routed ports, perform this procedure.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface* **Example:** `Device(config)# interface gigabitethernet1/1`	Enters interface configuration mode.
Step 3	**no switchport** **Example:** `Device(config-if)# no switchport`	Enters Layer 3 mode.

	Command or Action	Purpose
Step 4	**ip mtu** *bytes* **Example:** `Device(config-if)# ip mtu 850`	Changes the IPv4 MTU size. Valid values range from 832-1500.
Step 5	**ipv6 mtu** *bytes* **Example:** `Device(config-if)# ipv6 mtu 1280`	(Optional) Changes the IPv6 MTU size.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode, and returns to privileged EXEC mode.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves your entries in the configuration file.
Step 8	**show system mtu** **Example:** `Device# show system mtu`	Verifies your settings.

Configuration Examples for System MTU

Example: Configuring Protocol-Specific MTU

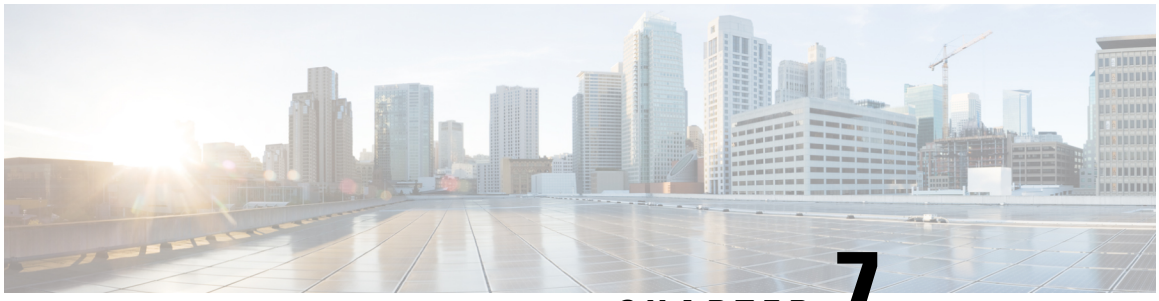
This example shows how you can configure protocol-specific MTU:

```
Device# configure terminal
Device(config)# interface gigabitethernet 1/1
Device(config-if)# ip mtu 900
Device(config-if)# ipv6 mtu 1286
Device(config-if)# end
```

Example: Configuring the System MTU

This example shows how you can configure the system MTU:

```
Device# configure terminal
Device(config)# system mtu 1600
Device(config)# exit
```



CHAPTER 7

Configuring Per-Port MTU

Restrictions for Per-Port MTU

- Per-Port MTU cannot be configured on SVL links.
- Members of a port channel cannot be configured with Per-Port MTU, they derive their MTU from the port-channel MTU configuration.
- Do not configure the per-port MTU value while the traffic is flowing.

Information About Per-Port MTU

You can configure the MTU size for all interfaces on a device at the same time using the **system mtu** command. The default maximum transmission unit (MTU) size for frames received and transmitted on all interfaces is 1500 bytes. The **system mtu** command is a global command and does not allow MTU to be configured at a port level. You can configure Per-Port MTU. Per-Port MTU will support port level and port channel level MTU configuration. With Per-Port MTU you can set different MTU values for different interfaces as well as different port channel interfaces.

Per-port MTU can be configured in the range of 1500-9198 bytes.

Once the Per-Port MTU value has been configured on a port, the protocol-specific MTU for that port is also changed to the Per-Port MTU value. When Per-Port MTU is configured on a port, you can still configure protocol-specific MTU on the interface in the range from 256 to Per-Port MTU value.

If the Per-Port MTU is disabled, the MTU for the port will revert to the system MTU value.

You can view the Per-Port MTU configurations on an interface using the **show interface mtu** command.

The following are expected behaviour if the Per-Port MTU configuration is changed on any interface:

- The interface flaps if the port-channel is in PAgP or LACP mode.
- The interface does not flap if the port channel is in the **on** mode.
- The interface does not flap if the interface is not a port channel.

You can disable Per-Port MTU by using the **no** form of the **mtu***bytes* command in the interface configuration mode.

Configuring Per-Port MTU

Follow these steps to change the MTU size for switched packets on a particular port of an interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface***type* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Configures the interface and enters interface configuration mode.
Step 4	**mtu***bytes* **Example:** `Device(config-if)# mtu 6666`	Configures the MTU size for a particular port on the interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode, and returns to privileged EXEC mode.

Example: Configuring Per-Port MTU

This example shows how to configure Per-Port MTU on an interface:

```
Device# configure terminal
Device(config)# interface GigabitEthernet 1/1
Device(config-if)# mtu 6666
Device(config-if)# end
```

Example: Verifying Per-Port MTU

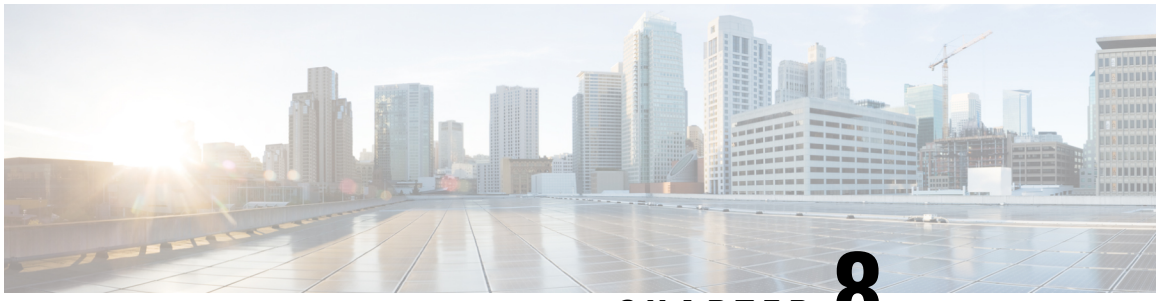
This example shows how to verify Per-Port MTU on an interface using the **show interface mtu** command:

```
Device# show interface mtu

Port           Name               MTU
Fo2/5/0/19                        1500
Fo2/5/0/20                        6666
Fo2/5/0/21      ixia_7_21         1500
```

Example: Disabling Per-Port MTU

This example shows how to disable Per-Port MTU on an interface:

```
Device# configure terminal
Device(config)# interface GigabitEthernet 1/1
Device(config-if)# no mtu
Device(config-if)# end
```



CHAPTER 8

Configuring Power over Ethernet

Information About Power over Ethernet

Power over Ethernet (PoE) is a technique for delivering DC power to devices over copper Ethernet cabling, eliminating the need for separate power supplies and outlets. Using PoE can improve flexible options for placing Ethernet end devices, and reduce the time and expense of installing electrical cabling.

A PoE-capable switch port automatically supplies power to one of these connected devices if the switch senses that there is no power in the circuit:

- An IEEE standard powered device, such as a new Cisco IP phone.
- An IEEE 802.3af-compliant powered device, which can receive up to 15.4 W of DC power
- An IEEE 802.3at-compliant powered device, which can receive up to 30 W of DC power
- An IEEE 802.3bt-compliant powered device, which can receive up to 90 W of DC power

A powered device can receive redundant power when it is connected to a PoE switch port and to an AC power source. The device does not receive redundant power when it is only connected to the PoE port.

Types of PoE

There are two types of PoE: PoE+, and Universal Power over Ethernet Plus (UPoE+). PoE+ delivers 30 W to a port while UPoE+ delivers 90 W to a port. However, both PoE+ and UPoE+ support lower wattages.

PoE features

- **PoE power policing:**When power policing is enabled, the device polices power usage by comparing the real-time power consumption with the maximum power allocated to the device.

  For more information, see the section Power Monitoring and Power Policing.

- **Perpetual PoE:** Perpetual PoE provides uninterrupted power to a connected powered device even when the power sourcing equipment switch is booting.

  For more information, see the section Configuring Perpetual PoE and Fast POE.

- **Fast PoE:** Fast PoE enables the quick start of PoE power after a system power loss and recovery. If Fast PoE is enabled, status of PoE ports is stored in flash so that if there is a power loss and recovery, ports can be powered on as quickly as possible.

Powered-Device Detection and Initial Power Allocation

The switch detects an IEEE-compliant powered device when the PoE-capable port is in the no-shutdown state, PoE is enabled (the default), and the connected device is not powered by an AC adapter.

After device detection, the switch determines the device's power requirements based on its type:

- The initial power allocation is the maximum amount of power that a powered device requires. The switch initially allocates this amount of power when it detects and powers the powered device. Because the switch receives CDP messages from the powered device, and because the powered device negotiates power levels with the switch through CDP power-negotiation messages, the initial power allocation might be adjusted.
- The switch classifies the detected IEEE device within a power consumption class. Based on the available power in the power budget, the switch determines if a port can be powered. The following table lists these levels.

Power Consumption Class	Maximum Power Level Required from Device
0 (class status unknown)	15.4 W
1	4 W
2	7 W
3	15.4 W
4	30 W
5	45 W
6	60 W
7	75 W
8	90 W

The following list shows the power consumption class and maxim power level required for the switches:

- E-3505-8P3S and IE-3500H-14P2T: Supports power consumption classes 1 through 4 (4 W to 30 W).
- IE3500H mGig 4PPoE switches (IE-3500H-12P2MU2X):
  - 12 GE downlink ports support power consumption classes 1 through 4 (4 W to 30 W).
  - Two 2.5 mGig downlink ports support power consumption classes 1 through 8 (4 W to 60 W).
- IE3500 mGig (IE-3500-8U3X): Support spower consumption classes 1 through 8 (4 W to 90 W).

The switch monitors and tracks requests for power and grants power only when it is available. The switch tracks the power budget (the amount of power available on the device for PoE). The switch also performs power-accounting calculations when a port is granted or denied power to keep the power budget up to date.

After power is applied to the port, the switch uses CDP to determine the CDP-specific power consumption requirement of the connected Cisco powered devices, which is the amount of power to allocate based on the CDP messages. The switch adjusts the power budget accordingly. Note that CDP does not apply to third-party PoE devices. The switch processes a request, and either grants or denies power. If the request is granted, the switch updates the power budget. If the request is denied, the switch ensures that the power to the port is turned off, generates a syslog message, and updates the LEDs. Powered devices can also negotiate with the switch for more power.

With third party IEEE powered devices use IEEE 802.3at or bt and LLDP power with medium-dependent interface (MDI) type, length, and value descriptions (TLVs) and power-via-MDI TLVs, for negotiating power up to 90 W. Cisco IEEE powered devices can use CDP or the IEEE 802.3 at or bt power-via-MDI power-negotiation mechanism to request power levels up to 30 W or 90 W.

If the switch detects a fault caused by an undervoltage, overvoltage, overtemperature, oscillator fault, or short-circuit condition, it turns off power to the port, generates a syslog message, and updates the power budget and LEDs.

Power Management Modes

The device supports these PoE modes:

- **Auto:** The auto mode is the default setting. The switch automatically detects if the connected device requires power. If the switch discovers a powered device connected to the port, and if the switch has enough power, it grants power, updates the power budget, and turns on power to the port on a first-come, first-served basis, and updates the LEDs. For LED information, see the hardware installation guide.

  If the switch has enough power for all the powered devices, they all come up. If enough power is available for all the powered devices connected to the switch, power is turned on to all the devices. If enough PoE is not available, or if a device is disconnected and reconnected while other devices are waiting for power, it cannot be determined which devices are granted or are denied power.

  If granting power exceeds the system's power budget, the switch denies power, ensures that power to the port is turned off, generates a syslog message, and updates the LEDs. After power is denied, the switch periodically rechecks the power budget and continues to attempt to grant the request for power.

  If a device that is being powered by the switch is then connected to wall power, the switch might continue to power the device. The switch might continue to report that it is still powering the device irrespective of whether the device is being powered by the switch or receiving power from an AC power source.

  If a powered device is removed, the switch automatically detects the disconnect and removes power from the port. You can connect a nonpowered device without damaging it.

  You can specify the maximum wattage that is allowed on the port. If the IEEE class maximum wattage of the powered device is greater than the configured maximum value, the switch does not provide power to the port. If the switch powers a powered device, but the powered device later requests, through CDP or LLDP messages, more than the configured maximum value, the switch removes power to the port. The power that was allocated to the powered device is reclaimed into the global power budget. If you do not specify a wattage, the switch delivers the maximum value. Use the auto setting on any PoE port.

- **Static:** The switch preallocates power to the port (even when no powered device is connected) and guarantees that power will be available for the port. The switch allocates the port-configured maximum wattage, and the amount is never adjusted through the IEEE class or by CDP messages from the powered device. Because power is preallocated, any powered device that uses less than or equal to the maximum wattage, is guaranteed to be powered when it is connected to the static port. The port no longer participates in the first-come, first-served model.

  However, if the powered device's IEEE class is greater than the maximum wattage, the switch does not supply power to it. If the switch learns through CDP messages that the powered device is consuming more than the maximum wattage, the switch shuts down the powered device.

  If you do not specify a wattage, the switch preallocates the maximum value. The switch powers the port only if it discovers a powered device.

- **Never:** The switch disables powered-device detection and never powers the PoE port even if an unpowered device is connected. Use this mode only when you want to make sure that power is never applied to a PoE-capable port, making the port a data-only port.

For most situations, the default configuration (**auto** mode) works well, providing plug-and-play operation. No further configuration is required. However, configure a PoE port to make it data only, or to specify a maximum wattage to disallow high-power powered devices on a port.

Power Monitoring and Power Policing

When policing of the real-time power consumption is enabled, the device takes action when a powered device consumes more power than the maximum amount allocated, which is also referred to as the *cutoff-power value*.

When PoE is enabled, the device senses and monitors the real-time power consumption of the connected powered device. This is called *power monitoring* or *power sensing*. The device also polices the power usage with the *power policing* feature.

Power monitoring is backward-compatible with Cisco intelligent power management and CDP-based power consumption. It works with these features to ensure that the PoE port can supply power to a powered device.

The device senses the real-time power consumption of the connected device as follows:

1. The device monitors the real-time power consumption by individual ports.
2. The device records the power consumption, including peak power usage, and reports this information through the CISCO-POWER-ETHERNET-EXT-MIB.
3. If power policing is enabled, the device polices power usage by comparing the real-time power consumption with the maximum power allocated to the device. The maximum power consumption is also referred to as the *cutoff power* on a PoE port.

   If the device uses more than the maximum power allocation on the port, the device can either turn off the power to the port, or can generate a syslog message and update the LEDs (the port LED is now blinking amber) while still providing power to the device based on the device configuration. By default, power-usage policing is disabled on all the PoE ports.

   If error recovery from the PoE error-disabled state is enabled, the device automatically takes the PoE port out of the error-disabled state after the specified amount of time.

   If error recovery is disabled, you can manually re-enable the PoE port by using the **shutdown** and **no shutdown** interface configuration commands.

4. If policing is disabled, no action occurs when the powered device consumes more than the maximum power allocation on the PoE port, which could adversely affect the device.

Power Consumption Values

You can configure the initial power allocation and the maximum power allocation on a port. However, these values are the configured values that determine when the device should turn on or turn off power on the PoE port. The maximum power allocation is not the same as the actual power consumption of the powered device. The actual cutoff power value that the device uses for power policing is not equal to the configured power value.

When power policing is enabled, the device polices the power usage *at the switch port*, where the power consumption is greater than that by the device. When you manually set the maximum power allocation, you must consider the power loss over the cable from the switch port to the powered device. The cutoff power is the sum of the rated power consumption of the powered device and the worst-case power loss over the cable.

We recommend that you enable power policing when PoE is enabled on your device. For example, for a Class 1 device, if policing is disabled and you set the cutoff-power value by using the **power inline auto max 6300** interface configuration command, the configured maximum power allocation on the PoE port is 6.3 W (6300 mW). The device provides power to the connected devices on the port if the device needs up to 6.3 W. If the CDP power-negotiated value or the IEEE classification value exceeds the configured cutoff value, the device does not provide power to the connected device. After the device turns on the power on the PoE port, the device does not police the real-time power consumption of the device, and the device can consume more power than the maximum allocated amount, which could adversely affect the device and the devices connected to the other PoE ports.

Universal Power Over Ethernet

Universal Power Over Ethernet (UPOE) technology extends the IEEE 802.3bt at PoE standard to provide the capability to source up to 90 W of power (refer to the datasheet for the specific power available for each PID) over standard Ethernet cabling infrastructure (Class D or better) by using the spare pair of an RJ-45 cable (wires 4,5,7,8) with the signal pair (wires 1,2,3,6). Power on the spare pair is enabled when the switch port and end device mutually identify themselves as UPOE-capable using CDP or LLDP and the end device's requests for power to be enabled on the spare pair. When the spare pair is powered, the end device can negotiate up to 90 W of power from the switch using CDP or LLDP.

If the end device supports detection and classification on both signal and spare pairs, but does not support the CDP or LLDP extensions required for UPOE, a 4-pair forced mode configuration automatically enables power on both signal and spare pairs from the switch port.

If the Single Signature PD provides a valid detection on the spare pair, then the switch port starts to deliver power to the spare pair as well.

Note When the port is denied power, the class value is shown in the **upoe** command output, not in the base **poe** command.

How to Configure PoE and UPOE

The following tasks describe how you can configure PoE and UPOE.

Use the global configuration command **power inline wattage max** *max-wattage* to configure the PoE budget of DIN rail switches. Limiting the PoE budget prevents overdrawing power and exceeding the capacity of the power source. For more information, see Configure PoE budget, on page 71

Configuring a Power Management Mode on a PoE Port

Note When you make PoE configuration changes, the port that are being configured drops power. Depending on the new configuration, the state of the other PoE ports and the state of the power budget, the port might not be powered up again. For example, port 1 is in the auto and on state, and you configure it for static mode. The device removes power from port 1, detects the powered device, and repowers the port. If port 1 is in the auto and on state, and you configure it with a maximum wattage of 10 W, the device removes power from the port and then redetects the powered device. The device repowers the port only if the powered device is a class 1, class 2.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the physical port to be configured, and enters interface configuration mode.
Step 4	**power inline** {**auto** [**max** *max-wattage*] | **never** | **static** [**max** *max-wattage*] | **consumption** *milli-watts-consumption* } **Example:** `Device(config-if)# power inline auto`	Configures the PoE mode on the port. The following are the keywords: • **auto**: Enables detection of powered devices. If enough power is available, automatically allocates power to the PoE port after device detection. This is the default setting. **Note** • The power inline auto max command should be used only to adjust the port's cut-off power if the connected powered device (PD) does not support Layer 1 classification via

	Command or Action	Purpose
		LLDP/CDP and requires a manual increase in power. • Configure the power inline auto max value to the maximum of 30 W for IEEE 802.3at or 802.3af PDs. The power inline auto max value should exceed 30W only when an IEEE 802.3bt compliant Dual Signature PD or a Class 5 (or higher) Single Signature PD is connected to the port. • **max** *max-wattage*: Limits the power allowed on the port. If no value is specified, the maximum is allowed. • **never**: Disables device detection and power to the port. **Note** If a port has a Cisco-powered device connected to it, do not use the **power inline never** command to configure the port. A false link-up can occur, placing the port in the error-disabled state. • **static**: Enables detection of powered devices. Preallocate (reserve) power for a port before the device discovers the powered device. The device reserves power for this port even when no device is connected, and guarantees that power will be provided upon device detection. **Note** • The power inline static max command should be used only to adjust the port's cut-off power if the connected powered device (PD) does not support Layer 1 classification via LLDP/CDP and requires a manual increase in power. • Configure the power inline static max value to the maximum of 30 W for IEEE 802.3at or 802.3af PDs. The power inline static max wattage value should exceed 30W only when an IEEE 802.3bt compliant

	Command or Action	Purpose
		Dual-Signature PD or a Class 5 (or higher) Single-Signature PD is connected to the port. • **consumption**: Sets the PoE consumption (in mW) of the powered device connected to a specific interface. The power consumption can range from 4000 to 90000 mW. Use the **no power inline consumption** command to return to the default settings. The **power inline consumption** *wattage* command allows you to override the default power requirement defined by the IEEE classification. By doing so, any difference between the IEEE-mandated power and the actual power required by the device is reclaimed into the global power budget. This reclaimed power can then be allocated to additional devices, enabling you to extend and utilize the switch power budget more efficiently. **Note** The **power inline consumption** *wattage* command is not supported for Dual Signature PDs. The device allocates power to a port configured in static mode before it allocates power to a port configured in auto mode.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show power inline** **Example:** `Device# show power inline`	Displays the PoE status for a device.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configure PoE budget

Use this task to configure the PoE budget of DIN railswitches.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**power inline wattage max** *max-wattage* **Example:** `Device(config-if)# power inline wattage max 360`	Configures the max-wattage power allowed on the port. If no value is specified, the default max-wattage value is 125 W. Refer to the datasheet for the max-wattage power available for each switch variant.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring Power Policing

By default, the device monitors the real-time power consumption of connected powered devices. You can configure the device to police the power usage. By default, policing is disabled.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the physical port to be configured, and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**power inline police** [**action**{**log** | **errdisable**}] **Example:** `Device(config-if)# power inline police`	Configures the device to take one of these actions if the real-time power consumption exceeds the maximum power allocation on the port: • **power inline police**: Shuts down the PoE port, turns off power to it, and puts it in the error-disabled state. **Note** You can enable error detection for the PoE error-disabled cause by using the **errdisable detect cause inline-power** global configuration command. You can also enable the timer to recover from the PoE error-disabled state by using the **errdisable recovery cause inline-power interval** *interval* global configuration command. • **power inline police action errdisable**: Turns off power to the port if the real-time power consumption exceeds the maximum power allocation on the port. • **power inline police action log**: Generates a syslog message while still providing power to the port. If you do not enter the **action log** keywords, the default action shuts down the port and puts the port in the error-disabled state.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode, and returns to global configuration mode.
Step 6	Use one of the following: • **errdisable detect cause inline-power** • **errdisable recovery cause inline-power** • **errdisable recovery interval** *interval* **Example:** `Device(config)# errdisable detect cause inline-power` `Device(config)# errdisable recovery cause inline-power` `Device(config)# errdisable recovery interval 100`	(Optional) Enables error recovery from the PoE error-disabled state, and configures the PoE recovery mechanism variables. By default, the recovery interval is 300 seconds. **interval** *interval*: Specifies the time in seconds, to recover from the error-disabled state. The range is 30 to 86400.

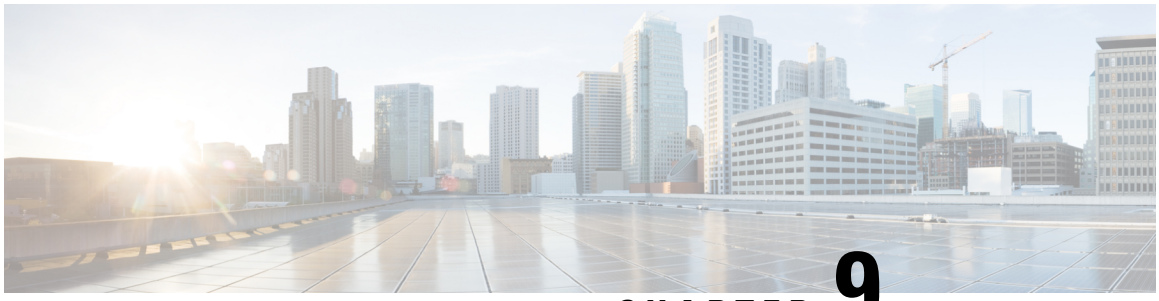
	Command or Action	Purpose
Step 7	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 8	Use one of the following: • **show power inline police** • **show errdisable recovery** **Example:** `Device# show power inline police` `Device# show errdisable recovery`	Displays the power-monitoring status, and verifies the error recovery settings.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring Power Status

Use the following **show** commands to monitor and verify the PoE configuration.

Table 10: show Commands for Power Status

Command	Purpose
show power inline police	Displays power-policing data.



CHAPTER 9

Configuring Perpetual PoE and Fast POE

Restrictions for Perpetual and Fast PoE

The following restrictions apply to perpetual and fast PoE:

- Configuration of Fast PoE or Perpetual PoE has to be done before physically connecting any endpoint. Alternatively do a manual shut/no-shut of the ports drawing power.
- Power to the ports will be interrupted in case of PSE firmware upgrade and ports will be back up immediately after the upgrade.
- The CREE light powered device (PD) may flap at regular intervals if not configured with IP assigned from the DHCP server.

Information About Perpetual PoE

Perpetual PoE provides uninterrupted power to connected powered device even when a power sourcing equipment (PSE) switch is starting after a reload from executing the Cisco IOS software **reload** command.

Caution Power to the ports will be interrupted in case of M3 or PSE firmware upgrade, and power to the ports will be backed up after Cisco IOS software starts.

Fast POE

This feature switches on power without waiting for IOS to boot up. When **poe-ha** is enabled on a particular port, the switch on a recovery after power failure, provides power to the connected endpoint devices within short duration before even the IOS forwarding starts up.

Configuring Perpetual and Fast PoE

To configure perpetual and Fast PoE, perform the following steps.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the physical port to be configured, and enters interface configuration mode.
Step 4	**power inline port perpetual-poe-ha** **Example:** `Device(config-if)# power inline port perpetual-poe-ha`	Configures perpetual PoE. When you configure perpetual PoE on a port connected to a powered device, the powered device remains powered on during reload.
Step 5	**power inline port poe-ha** **Example:** `Device(config-if)# power inline port poe-ha`	Configures Fast PoE. When you configure Fast PoE, if the switch is power cycled, PD device powers on within 50-60 seconds of plugging into a power source without waiting for IOS to boot up.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

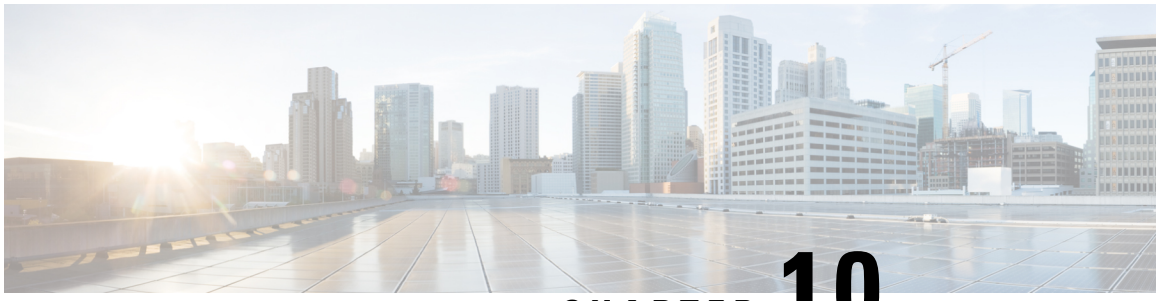
Example: Configuring Perpetual and Fast PoE

This example shows how you can configure perpetual PoE on a switch:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet1/1
Device(config-if)# power inline port perpetual-poe-ha
Device(config-if)# end
```

This example shows how you can configure fast PoE on the switch:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet1/1
Device(config-if)# power inline port poe-ha
Device(config-if)# end
```



CHAPTER 10

Configuring Auto SmartPorts

Restrictions for Auto SmartPorts

- Although Auto SmartPort detects the Cisco switch it does not invoke the event trigger automatically. The event trigger needs to be manually invoked to map the switch to macros.

  The **no macro auto global processing** command disables the Auto Smartport only. To disable the device classifier, use the **no device classifier** command.

- In a scenario where the user is authenticating for clients using the ASP macro and the macro includes commands that may trigger a session teardown or an internal configuration change, we observe that after authentication, the MAC address gets stuck in the drop state. The following are recommended workarounds to avoid this situation:

  - If the macro contains authentication commands, such as **authentication event server dead action authorize vlan** *vlan-id* and **authentication event no-response action authorize vlan** *vlan-id*, remove the commands from the macro and configure them directly on the interface.

  - If the macro contains the **switchport access vlan***vlan-id* command, use the Dynamic VLAN from the AAA server instead of configuring the VLAN via the macro.

Information about Auto SmartPorts

Auto SmartPort macros dynamically configure ports based on the device type detected on the port. When the switch detects a new device on a port, it applies the appropriate Auto SmartPorts macro. When a link-down event occurs on the port, the switch removes the macro. For example, when you connect a Cisco IP phone to a port, Auto SmartPorts automatically applies the Cisco IP phone macro. The Cisco IP phone macro enables quality of service (QoS), security features, and a dedicated voice VLAN to ensure proper treatment of delay-sensitive voice traffic.

Auto SmartPorts uses event triggers to map devices to macros. The most common event triggers are based on Cisco Discovery Protocol (CDP) messages received from connected devices. The detection of a device (Cisco IP phone, Cisco wireless access point, or Cisco router) invokes an event trigger for that device.

Link Layer Discovery Protocol (LLDP) is used to detect devices that do not support CDP. Other mechanisms used as event triggers include the 802.1X authentication result and MAC-address learned.

System built-in event triggers exist for various devices based mostly on CDP and LLDP messages and some MAC address. These triggers are enabled as long as Auto SmartPort is enabled.

You can configure user-defined trigger groups for profiles and devices. The name of the trigger group is used to associate a user-defined macro.

Auto SmartPort Macros

The Auto SmartPort macros are groups of CLI commands. Detection of devices on a port triggers the application of the macro for the device. System built-in macros exist for various devices, and, by default, system built-in triggers are mapped to the corresponding built-in macros. You can change the mapping of built-in triggers or macros as needed.

A macro basically applies or removes a set of CLIs on an interface based on the link status. In a macro, the link status is checked. If the link is up, then a set of CLIs is applied; if the link is down, the set is removed (the no format of the CLIs are applied). The part of the macro that applies the set of CLIs is termed macro. The part that removes the CLIs (the no format of the CLIs) are termed antimacro.

When a device is connected to an Auto SmartPort, if it gets classified as a lighting end point, it invokes the event trigger **CISCO_LIGHT_EVENT**, and the macro **CISCO_LIGHT_AUTO_SMARTPORT** is executed.

Commands run by CISCO_LIGHT_AUTO_SMARTPORT

When the macro is executed, it runs a series of commands on the switch.

The commands that are executed by running the macro **CISCO_LIGHT_AUTO_SMARTPORT** are:

- switchport mode access
- switchport port-security violation restrict
- switchport port-security mac-address sticky
- switchport port-security
- power inline port poe-ha
- storm-control broadcast level 50.00
- storm-control multicast level 50.00
- storm-control unicast level 50.00
- spanning-tree portfast
- spanning-tree bpduguard enable

Enabling Auto SmartPort

Note Auto SmartPorts are disabled by default.

To disable Auto SmartPort macros on a specific port, use the **no macro auto global processing** interface command before enabling Auto SmartPort globally.

To enable Auto SmartPort globally, use the **macro auto global processing** global configuration command.

To enable an Auto SmartPort, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**device classifier** **Example:** `Device(config)# device classifier`	Enables the device classifier. Use **no device classifier** command to disable the device classifier.
Step 4	**macro auto global processing** **Example:** `Device(config)# macro auto global processing`	Enables Auto SmartPorts on the switch globally. Use **no macro auto global processing** command to disable Auto SmartPort globally.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

How to Configure Auto SmartPorts

The following section provides information about how to configure auto smartports.

Note Follow these guidelines when you are configuring Auto SmartPort Macros, performing active standby sync and configuring reload from primary to standby:

- Make sure there is no extra space in the configuration.
- Do not add extra parenthesis and tab in the configuration.
- Ensure that you do not use enter keyword more than required while configuring.

Configuring Mapping Between Event Triggers and Built-in Macros

To map an event trigger to a built-in macro, perform this task:

Before you begin

You need to enable Auto SmartPort macros globally. You need to perform this task when a Cisco switch is connected to the Auto SmartPort.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**macro auto execute** *event trigger* **builtin** *built-in macro name* **Example:** `Device(config)# macro auto execute CISCO_SWITCH_EVENT builtin CISCO_SWITCH_AUTO_SMARTPORT`	Specifies a user-defined event trigger and a macro name. This action configures mapping from an event trigger to a built-in Auto Smartports macro.
Step 4	**macro auto trigger** *event trigger* **Example:** `Device(config)# macro auto trigger CISCO_SWITCH_EVENT`	Invokes the user-defined event trigger.

	Command or Action	Purpose
Step 5	**device** *device_ID* **Example:** `Device(config)# device cisco WS-C3560CX-8PT-S`	Matches the event trigger to the device identifier.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show shell triggers** **Example:** `Device# show shell triggers`	Displays the event triggers on the switch.
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuration Examples for Auto SmartPorts

The following sections provide configuration examples for Auto SmartPorts.

Example: Enabling Auto SmartPorts

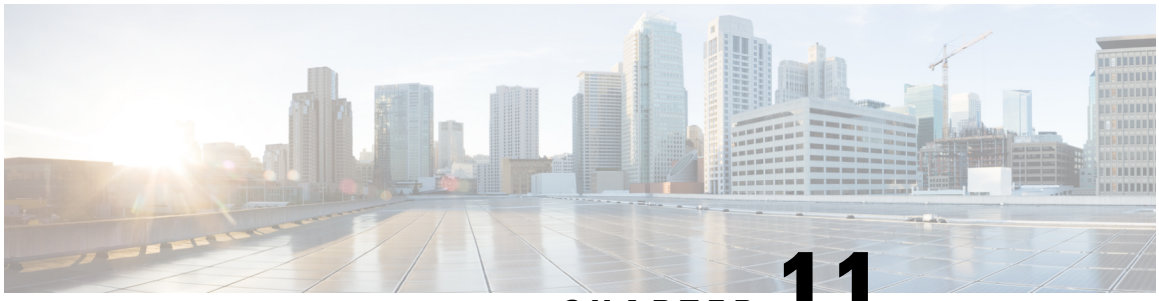
The following example shows how you can enable an Auto SmartPort.

```
Device> enable
Device# configure terminal
Device(config)# device classifier
Device(config)# macro auto global processing
Device(config)# end
```

Example: Configuring Mapping Between Event Triggers and Built-In Macros

The following example shows how you can configure mapping between event triggers and built-in macros:

```
Device> enable
Device# configure terminal
Device(config)# macro auto execute CISCO_SWITCH_EVENT builtin CISCO_SWITCH_AUTO_SMARTPORT
Device(config)# macro auto trigger CISCO_SWITCH_EVENT
Device(config)# device cisco WS-C3560CX-8PT-S
Device(config)# end
```



CHAPTER 11

Locate the switch on a Network

Locate a switch overview

The locate switch lets you easily find a specific switch on your network physically. The **locate-switch** command in the Command Line Interface (CLI) also helps you locate a required switch or switches on your network. This feature keeps the system LEDs illuminated on a particular switch or switches for a predetermined duration. It is useful for identifying the required switch among many interconnected devices in a room.

Locate the switch on a network

You can find the physical location of a particular switch by using the **locate switch** command to activate the ALT_GREEN_RED color LED on the front panel of the switch.

Procedure

Step 1 Login to the switch console and enter the login credentials.

Step 2 Use the **locate-switch** command to locate the system LEDs on the switch.

Example:

```
Switch#locate-switch switch active
```

Step 3 Enter the time between 9 to 255 (in seconds) for the LED of the switch to stay active.

Example:

```
Switch#locate-switch switch active 255
```

Step 4 Enter the time as 0 seconds to stop the LED blinking.

Example:

```
Switch#locate-switch switch active 0
```

Note

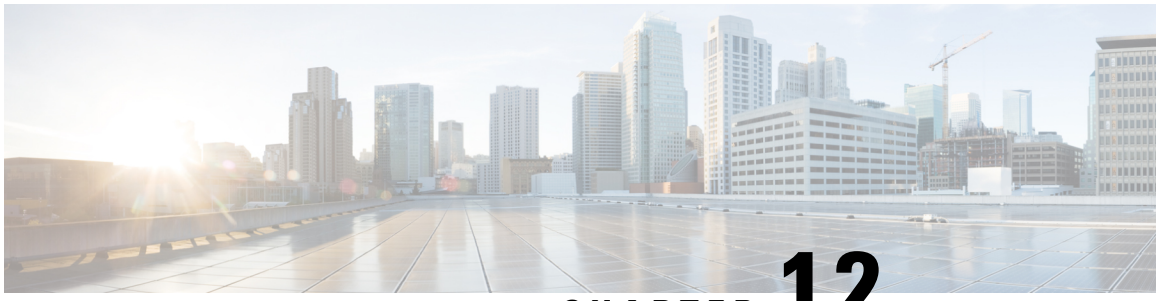
Do not use the Standby feature on IE3500 switches, as it is a standalone device and not a stacked switch.

Verify Switch Location

Verify the location of the switch by physically checking the switch LED to indicate the required switch on your network setup. The CLI also helps you locate a required switch on your network.

The CLI displays the switch location on a show command:

```
Switch#show hardware led

Current Mode: STATUS

SYSTEM: ALT_GREEN_RED
EXPRESS-SETUP: ALT_GREEN_RED
DC-A: GREEN
DC-B: RED
ALARM-OUT: ALT_GREEN_RED
ALARM-IN1: ALT_GREEN_RED
ALARM-IN2: ALT_GREEN_RED
POE: ALT_GREEN_RED
STATUS: (28) Gi1/1:BLACK Gi1/2:FLASH_GREEN Gi1/3:BLACK Gi1/4:BLACK Gi1/5:ACT_GREEN Gi1/6:BLACK
 Gi1/7:BLACK Gi1/8:BLACK Gi1/9:BLACK Gi1/10:BLACK Gi1/11:ACT_GREEN Gi2/1:ACT_GREEN Gi2/2:BLACK
 Gi2/3:BLACK Gi2/4:ACT_GREEN Gi2/5:BLACK Gi2/6:ACT_GREEN Gi2/7:BLACK Gi2/8:BLACK Gi2/9:BLACK
 Gi2/10:BLACK Gi2/11:BLACK Gi2/12:BLACK Gi2/13:BLACK Gi2/14:BLACK Gi2/15:BLACK
Gi2/16:ACT_GREEN
```



CHAPTER 12

Switch Alarms

Information about switch alarms

The switch software monitors switch conditions on a per-port or a switch basis. If the conditions present on the switch or a port do not match the set parameters, the switch software triggers an alarm or a system message.

By default, the switch software sends the system messages to a system message logging facility, or a *syslog* facility. You can also configure the switch to send Simple Network Management Protocol (SNMP) traps to an SNMP server.

External alarms

The Cisco IE3500 Rugged Series Switches support two alarm inputs and one alarm output, while the Cisco IE3500H Heavy Duty Series Switches support one alarm input and one alarm output.

The alarm input circuit is designed to sense if a dry contact is open or closed relative to the Alarm-In reference pin. The Alarm_Out is a relay with Normally Open and Normally Closed contacts.

The switch software is configured to detect faults which are used to energize the relay coil and change the state on both of the relay contacts. Normally open contacts close and normally closed contacts open.

- **Open** means that the normal condition has current flowing through the contact (normally closed contact). The alarm is generated when the current stops flowing.
- **Closed** means that no current flows through the contact (normally open contact). The alarm is generated when current does flow.

Note Software can program the Alarm_In to trigger an alarm with either Open or Closed setting.

You can set the alarm severity to major, minor, or none. The severity is included in the alarm message and also sets the LED color when the alarm is triggered. The LED is red for a minor alarm and blinking red for a major alarm. If not set, the default alarm severity is minor.

For detailed information about the alarm connector, LEDs, alarm circuit and wiring installation, alarm ratings and ports, see the *Hardware Installation Guide*.

Power supply alarms

Cisco IE3500 Rugged Series Switches

The Cisco IE3500 Rugged Series Switches have two power supply slots that contain DC power supplies. One input is used for system operation, while the second input is optional for redundancy, and the system cannot share power between these two inputs. The switch LEDs display the status and type of power supplied to each slot. The DC-power supplies have two DC inputs (DC A and DC B).

The default power supply configuration is to have one power supply installed in slot 1 and the software configured for **no power-supply dual**. This suppresses any alarms triggered by not having two power supplies installed. When the switch is operating with two power supplies, we recommend you enter the **power-supply dual** global configuration command to trigger an alarm when one is missing or inoperable.

When the switch detects a power supply fault, it triggers an LED indicator and sends a system message. Power-supply alarm indications are sent when a power supply is missing, has no input, or has insufficient output. Some of these alarm conditions are configurable.

The Cisco IE3500H Heavy Duty Series Switches

The Cisco IE3500H Heavy Duty Series Switches support a single DC power inputs. The power supply operates within a voltage range of 85-264V AC or 20-110V DC, delivers a 54V DC output and provides 360W of total power at 70°C.

Note The Cisco IE3500H Heavy Duty Series Switches does not support power supply alarm.

Power-Supply-Missing Alarms

If you are operating the Cisco IE3500 Rugged Series Switch with a single power supply (DC), you can suppress any alarm conditions associated with a missing power supply. Entering the **no power-supply dual global** configuration command (the default) specifies that only one power supply is expected to be present. Then the switch does not generate an alarm that a power supply is missing. The **no power-supply dual** command

controls only the sending of messages about the absence of a second power supply or the absence of input to the second power supply. The software detects whether a power supply is present and if there is an input voltage. When there is input, the software can detect if there is output voltage.

If you operate the Cisco IE3500 Rugged Series Switches with two power supplies, enter the **power-supply dual** global configuration command to configure the switch to send a message when one power supply is missing.

Global status monitoring alarms

The switch processes alarms related to temperature and power supply conditions, referred to as global or facility alarms.

Table 11: Global status monitoring alarms

Alarm	Description
Power supply alarm	By default, the switch monitors a single power supply. If you configure a dual power supply, an alarm triggers if one power supply fails. You can configure the power supply alarm to be connected to the hardware relays. For more information, see the Configure power supply alarms, on page 94.
Temperature alarms	The switch contains one temperature sensor with a primary and secondary temperature setting. The sensor monitors the environmental conditions inside the switch. The primary and secondary temperature alarms can be set as follows: • The primary alarm is enabled automatically to trigger both at a low temperature, –4°F (–20°C) and a high temperature, 203°F (95°C). It cannot be disabled. By default, the primary temperature alarm is associated with the major relay. • The secondary alarm triggers when the system temperature is higher or lower than the configured high and low temperature thresholds. The secondary alarm is disabled by default. For more information, see the Configure switch temperature alarms, on page 96.
SD-Card	By default the alarm is disabled.

FCS error hysteresis threshold

The Ethernet standard calls for a maximum bit-error rate of 10^{-8}. The bit error-rate range is from 10^{-6} to 10^{-11}. The bit error-rate input to the switch is a positive exponent. If you want to configure the bit error-rate of 10^{-9}, enter the value 9 for the exponent. By default, the FCS bit error-rate is 10^{-8}.

You can set the FCS error hysteresis threshold to prevent the toggle of the alarm when the actual bit-error rate fluctuates near the configured rate. The hysteresis threshold is defined as the ratio between the alarm clear threshold to the alarm set threshold, expressed as a percentage value.

For example, if the FCS bit error-rate alarm value is configured to 10^{-8}, that value is the alarm set threshold. To set the alarm clear threshold at $5*10^{-10}$, the hysteresis, value *h*, is determined as follows:

h = alarm clear threshold / alarm set threshold

$h = 5*10^{-10} / 10^{-8} = 5*10^{-2} = 0.05 = 5$ percent

The FCS hysteresis threshold is applied to all ports on the switch. The allowable range is from 1 to 10 percent. The default value is 10 percent. For more information, see the Configure FCS error threshold, on page 99.

Port status monitoring alarms

The switch can also monitor the status of the Ethernet ports and generate alarm messages based on the alarms listed in port status monitoring alarms table. To save user time and effort, it supports changeable alarm configurations by using alarm profiles. You can create a number of profiles and assign one of these profiles to each Ethernet port.

Alarm profiles provide a mechanism for you to enable or disable alarm conditions for a port and associate the alarm conditions with one or both alarm relays. You can also use alarm profiles to set alarm conditions to send alarm traps to an SNMP server and system messages to a syslog server. The alarm profile defaultPort is applied to all interfaces in the factory configuration (by default).

Note You can associate multiple alarms to one relay or one alarm to both relays.

Port status monitoring alarms table given below lists the port status monitoring alarms and their descriptions and functions. Each fault condition is assigned a severity level based on the Cisco IOS System Error Message Severity Level.

Table 12: Port status monitoring alarms

Alarm List ID	Alarm	Description
1	Link Fault alarm	The switch generates a link fault alarm when problems with a port physical layer cause unreliable data transmission. A typical link fault condition is loss of signal or clock. The link fault alarm is cleared automatically when the link fault condition is cleared. The severity for this alarm is error condition, level 3.
2	Port not Forwarding alarm	The switch generates a port not-forwarding alarm when a port is not forwarding packets. This alarm is cleared automatically when the port begins to forward packets. The severity for this alarm is warning, level 4.
3	Port not Operating alarm	The switch generates a port not-operating alarm when a port fails during the startup self-test. When triggered, the port not-operating alarm is only cleared when the switch is restarted and the port is operational. The severity for this alarm is error condition, level 3.
4	FCS Bit Error Rate alarm	The switch generates an FCS bit error-rate alarm when the actual FCS bit error-rate is close to the configured rate. You can set the FCS bit error-rate by using the interface configuration CLI for each of the ports. For more information, see the Configure FCS error threshold, on page 99. The severity for this alarm is error condition, level 3.

Trigger alarm options

The switch supports these methods for triggering alarms:

- Configurable Relay

The switch is equipped with one independent alarm relay that can be triggered by alarms for global, port status and SD flash card conditions. You can configure the relay to send a fault signal to an external alarm device, such as a bell, light, or other signaling device. You can associate any alarm condition with the alarm relay. Each fault condition is assigned a severity level based on the Cisco IOS System Error Message Severity Level.

For more information on configuring the relay, see the Configure power supply alarms, on page 94.

- SNMP Traps

  SNMP is an application-layer protocol that provides a message format for communication between managers and agents. The SNMP system consists of an SNMP manager, an SNMP agent, and a management information base (MIB).

- The snmp-server enable traps command can be changed so that the user can send alarm traps to an SNMP server. You can use alarm profiles to set environmental or port status alarm conditions to send SNMP alarm traps. For more information, see the Enable SNMP traps, on page 102.

- Syslog Messages

  You can use alarm profiles to send system messages to a syslog server. For more information, see the Configure power supply alarms, on page 94.

Default switch alarm settings

Table 13: Default switch alarm settings

	Alarm	Default Setting
Global	Power supply alarm	Enabled in switch single power mode. No alarm. In dual-power supply mode, the default alarm notification is a system message to the console.
	Primary temperature alarm	Enabled for switch temperature range of 203°F (95°C) maximum to –4°F (–20°C) minimum. The primary switch temperature alarm is associated with the major relay.
	Secondary temperature alarm	Disabled.
	Output relay mode alarm	Normally deenergized. The alarm output has switched off or is in an off state.

	Alarm	Default Setting
Port	Link fault alarm	Disabled on all interfaces.
	Port not forwarding alarm	Disabled on all interfaces.
	Port not operating alarm	Enable on all interfaces.
	FCS bit error rate alarm	Disabled on all interfaces.

Configure switch alarms

Configure external alarms

Use this task to configure the alarms on Cisco IE3500 Series Switch

Note The Cisco IE3500H Heavy Duty Series Switches supports one alarm inputs and one alarm output.

Procedure

	Command or Action	Purpose
Step 1	Switch#**configure terminal**	Enter global configuration mode
Step 2	Switch(config)#**alarm contact** *contact-number* **description** *string* **Example:** `Switch(config)#alarm contact 1 description door sensor`	(Optional) Configures a description for the alarm contact number. • The *contact-number* value is from 1 to 4. • The description string is up to 80 alphanumeric characters in length and is included in any generated system messages.
Step 3	Switch(config)#**alarm contact** {**contact-number** | **all**} {**severity** {*major* / *minor* / *none*} | {*closed* / *open*}} **Example:** `Switch(config)#alarm contact 1 severity major`	Configures the trigger and severity for an alarm contact number or for all contact numbers. • Enter a contact number (1 to 4) or specify that you are configuring **all** alarms. • For severity, enter **major**, **minor** or **none**. If you do not configure a severity, the default is **minor**. • For trigger, enter **open** or **closed**. If you do not configure a trigger, the alarm is triggered when the circuit is **closed**.

	Command or Action	Purpose
Step 4	Switch(config)# **alarm relay-mode energized**	(Optional) Configures the output relay mode to energized.
Step 5	Switch#**show env alarm-contact** **Example:** • The given output of the **show env alarm-contact** command is from the Cisco IE3500 Rugged Series Switch. `Switch#show env alarm-contact` `ALARM CONTACT 1` `Status: not asserted` `Description: door sensor` `Severity: major` `Trigger: closed` `ALARM CONTACT 2` `Status: not asserted` `Description: external alarm contact 2` `Severity: minor` `Trigger: closed` • The given output of the **show env alarm-contact** command is from the Cisco IE3500H Heavy Duty Series Switch. `Switch#sh env alarm-contact` `Switch: 1` `ALARM CONTACT 1` `Status: not asserted` `Description: external alarm contact 1` `Severity: minor` `Trigger: closed`	(Optional) Verifies the configured alarm contacts.
Step 6	Switch(config)#**copy running-config startup-config**	Saves your entries in the configuration file.

Configure power supply alarms

Use this task to configure the power supply alarms on the Cisco IE3500 Rugged Series Switches.

Note The Cisco IE3500H Heavy Duty Series Switches does not support power supply alarm.

Procedure

	Command or Action	Purpose
Step 1	Switch(config)#**power-supply dual**	Configures dual power supplies.

	Command or Action	Purpose
Step 2	Switch(config)#**alarm facility power-supply disable**	Disables the power supply alarm.
Step 3	Switch(config)#**alarm facility power-supply relay major** **Example:** Switch(config)#**alarm contact 1 severity major**	(Optional) Associates the power supply alarm to the relay.
Step 4	Switch#**alarm facility power-supply notifies**	(Optional) Sends power supply alarm traps to an SNMP server.
Step 5	Switch#**alarm facility power-supply syslog**	(Optional) Sends power supply alarm traps to a syslog server.
Step 6	Switch#**show facility-alarm status** **Example:** `Switch#show env power` `POWER SUPPLY A is DC OK` `POWER SUPPLY B is DC FAULTY <--`	Displays the switch power status
Step 7	Switch#**show facility-alarm status** **Example:** `Switch#show facility-alarm status` `Source Severity Description Relay Time` `Switch MAJOR 5 Redundant Pwr missing or failed NONE Mar 01 1993 00:23:52`	Displays all generated alarms for the switch.
Step 8	Switch#**show alarm settings** **Example:** `Switch#show alarm settings` `Alarm relay mode: De-energized` `Power Supply` `Alarm Enabled` `Relay` `Notifies Disabled` `Syslog Enabled` `Temperature-Primary` `Alarm Enabled` `Thresholds MAX: 95C MIN: -20C` `Relay MAJ` `Notifies Enabled` `Syslog Enabled` `Temperature-Secondary` `Alarm Disabled` `Threshold` `Relay` `Notifies Disabled` `Syslog Disabled` `SD-Card` `Alarm Disabled` `Relay` `Notifies Disabled`	Verifies the configuration.

	Command or Action	Purpose
	Syslog Enabled Input-Alarm 1 Alarm Enabled Relay Notifies Disabled Syslog Enabled Input-Alarm 2 Alarm Enabled Relay Notifies Disabled Syslog Enabled	
Step 9	Switch(config)#**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Configure switch temperature alarms

Use this task to configure the temperature alarms on Cisco IE3500 Series Switch.

Procedure

	Command or Action	Purpose
Step 1	Switch(config)#**alarm facility temperature** {**primary** | **secondary**} **high** *threshold* **Example:** `Switch(config)#alarm facility temperature secondary high 45`	Configures the high temperature threshold value. **Note** The threshold range is from –238°F (–150°C) to 572°F (300°C).
Step 2	Switch(config)#**alarm facility temperature primary low** *threshold* **Example:** `Switch(config)#alarm facility temperature primary low 10`	Configures the low temperature threshold value. **Note** The threshold range is from –328°F (–200°C) to 482°F (250°C).
Step 3	Switch# **show alarm settings** **Example:** The given output of the **show alarm settings** command is from the Cisco IE3500H Heavy Duty Series Switch. `Switch#show alarm settings` `Alarm relay mode: Positive` `Temperature-Primary` `Alarm Enabled` `Thresholds MAX: 80C MIN: 0C` `Relay MAJ` `Notifies Enabled` `Syslog Enabled`	(Optional) Verifies the configuration.

	Command or Action	Purpose
	Temperature-Secondary Alarm Disabled Threshold Relay Notifies Disabled Syslog Disabled SD-Card Alarm Disabled Relay Notifies Disabled Syslog Enabled Input-Alarm 1 Alarm Enabled Relay Notifies Disabled Syslog Enabled PTP Alarm Disabled Relay Notifies Disabled Syslog Disabled HSR Alarm Disabled Relay Notifies Disabled Syslog Disabled DLR Alarm Disabled Relay Notifies Disabled Syslog Disabled	
Step 4	Switch(config)#**copy running-config startup-config**	Saves your entries in the configuration file.

Associate temperature alarms to a relay

By default, the primary temperature alarm is associated to the relay.

You can use the **alarm facility temperature** global configuration command to associate the primary temperature alarm to an SNMP trap, or a syslog message, or to associate the secondary temperature alarm to the relay, an SNMP trap, or a syslog message.

Note The single relay on the switch is called the major relay.

Use this task to associate temperature alarms to a relay on Cisco IE3500 Series Switch.

Procedure

Step 1 Switch(config)#**alarm facility temperature** {**primary** | **secondary**} **relay major**

Associates the primary or secondary temperature alarm to the relay.

Step 2 Switch(config)#**alarm facility temperature** {**primary** | **secondary**} **notifies**

Sends primary or secondary temperature alarm traps to an SNMP server.

Step 3 Switch(config)#**alarm facility temperature** {**primary** | **secondary**} **syslog**

Sends primary or secondary temperature alarm traps to a syslog server.

(Optional) Use the **no alarm facility temperature secondary** command to disable the secondary temperature alarm.

Step 4 Switch#**show alarm settings**

Example:

The output of the **show env alarm-contact** command is from the Cisco IE3500 Rugged Series Switch.

```
Switch#show alarm settings
Alarm relay mode: De-energized
Power Supply
Alarm Enabled
Relay
Notifies Disabled
Syslog Enabled
Temperature-Primary
Alarm Enabled
Thresholds MAX: 95C MIN: -20C
Relay MAJ
Notifies Enabled
Syslog Enabled
Temperature-Secondary
Alarm Disabled
Threshold
Relay
Notifies Disabled
Syslog Disabled
SD-Card
Alarm Disabled
Relay
Notifies Disabled
Syslog Enabled
Input-Alarm 1
Alarm Enabled
Relay
Notifies Disabled
Syslog Enabled
Input-Alarm 2
Alarm Enabled
Relay
Notifies Disabled
Syslog Enabled
```

Verifies the configuration.

Step 5 Switch(config)#**copy running-config startup-config** to

(Optional) Saves your entries in the configuration file.

Configure FCS bit error rate alarm

Configure FCS error threshold

Use this task to set the FCS bit error-rate alarm when the actual rate is close to the configured rate on Cisco IE3500 Series Switch.

Procedure

	Command or Action	Purpose
Step 1	Switch(config)#**interface** *interface-id* **Example:** `Switch(config)#interface gigabitethernet1/1`	Enters the desired interface and switch to interface configuration mode.
Step 2	Switch(config)#**fcs-threshold** *value* **Example:** `Switch(config)#fcs-threshold 10`	Sets the FCS error rate For *value*, the range is 6 to 11 to set a maximum bit error rate of 10^{-6} to 10^{-11}. By default, the FCS bit error rate is 10^{-8}.
Step 3	Switch#**show env alarm-contact** **Example:** The output of the **show env alarm-contact** command is from the Cisco IE3500H Heavy Duty Series Switch. `Switch#show env alarm-contact` `ALARM CONTACT 1` `Status: not asserted` `Description: door sensor` `Severity: major` `Trigger: closed` `ALARM CONTACT 2` `Status: not asserted` `Description: external alarm contact 2` `Severity: minor` `Trigger: closed`	(Optional) Verifies the configured alarm contacts.
Step 4	(Optional) Use the **copy running-config startup-config** to save your entries in the configuration file.	`Switch(config)#copy running-config startup-config`

Configure FCS error hysteresis threshold

The hysteresis setting prevents the toggle of an alarm when the actual bit error-rate fluctuates near the configured rate. The FCS hysteresis threshold is applied to all ports of a switch.

Use this task to set the error hysteresis threshold on Cisco IE3500 Series Switch.

Procedure

	Command or Action	Purpose
Step 1	Switch(config)#**alarm facility fcs-hysteresis** *percentage* **Example:** `Switch(config)#alarm facility fcs-hysteresis 10`	Sets the hysteresis percentage for the Cisco IE3500 Series Switch. For percentage, the range is 1 to 10. The default value is 10 percent
Step 2	Switch#**show env alarm-contact** **Example:** The output of the **show env alarm-contact** command is from the Cisco IE3500H Heavy Duty Series Switch. `Switch#show env alarm-contact` `ALARM CONTACT 1` `Status: not asserted` `Description: door sensor` `Severity: major` `Trigger: closed` `ALARM CONTACT 2` `Status: not asserted` `Description: external alarm contact 2` `Severity: minor` `Trigger: closed`	(Optional) Verifies the configured alarm contacts.
Step 3	Switch(config)#**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Configure alarm profiles

Use this task to create alarms profile on Cisco IE3500 Series Switch.

Procedure

	Command or Action	Purpose
Step 1	Switch(config)#**alarm-profile** *name* **Example:** `Switch(config)#alarm-profile fastE`	Creates the new profile or identifies an existing profile, and enter alarm profile configuration mode.

	Command or Action	Purpose
		When you create a new alarm profile, none of the alarms are enabled. **Note** The only alarm enabled in the *defaultPort* profile is the Port not operating alarm.
Step 2	Switch(config-alarm-profile)#**alarm** {**fcs-error** | **link-fault** | **not-forwarding** | **not-operating** **Example:** `Switch(config-alarm-profile)#alarm fcs-error`	(Optional) Adds or modifies alarm parameters for a specific alarm.
Step 3	Switch(config-alarm-profile)#**notifies** {**fcs-error** | **link-fault** | **not-forwarding** | **not-operating** **Example:** `Switch(config-alarm-profile)#notifies not-forwarding`	(Optional) Configures the alarm to send an SNMP trap to an SNMP server.
Step 4	Switch(config-alarm-profile)#**relay-major** {**fcs-error** | **link-fault** | **not-forwarding** | **not-operating** **Example:** `Switch(config-alarm-profile)#relay major link-fault`	(Optional) Configures the alarm to send an alarm trap to the relay.
Step 5	Switch(config-alarm-profile)#**syslog** {**fcs-error** | **link-fault** | **not-forwarding** | **not-operating** **Example:** `Switch(config-alarm-profile)#syslog not-forwarding`	(Optional) Configures the alarm to send an alarm trap to a syslog server.
Step 6	Switch#**show env alarm-contact** **Example:** The output of the **show env alarm-contact** command is from the Cisco IE3500H Heavy Duty Series Switch. `Switch#show env alarm-contact` `ALARM CONTACT 1` `Status: not asserted` `Description: door sensor` `Severity: major` `Trigger: closed` `ALARM CONTACT 2` `Status: not asserted` `Description: external alarm contact 2` `Severity: minor` `Trigger: closed`	(Optional) Verifies the configured alarm contacts.

	Command or Action	Purpose
Step 7	Switch(config)#**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Attach alarm profile to a specific port

Use this task to attach alarm profile to a specific port on Cisco IE3500 Series Switch.

Procedure

	Command or Action	Purpose
Step 1	Switch(config)#**interface** *port-interface*	Enters interface configuration mode.
Step 2	Switch(config)#**alarm-profile** *name* **Example:** `Switch(config)#alarm profile fastE`	Attaches the specified profile to the interface.
Step 3	Switch#**show env alarm-contact** **Example:** The output of the **show env alarm-contact** command is from the Cisco IE3500H Heavy Duty Series Switch. `Switch#show env alarm-contact` `ALARM CONTACT 1` `Status: not asserted` `Description: door sensor` `Severity: major` `Trigger: closed` `ALARM CONTACT 2` `Status: not asserted` `Description: external alarm contact 2` `Severity: minor` `Trigger: closed`	(Optional) Verifies the configured alarm contacts.
Step 4	Switch(config)#**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Enable SNMP traps

Use this task to enable SNMP traps on Cisco IE3500 Series Switch.

Procedure

	Command or Action	Purpose
Step 1	Switch(config)#**snmp-server enable traps alarms**	Enables the switch to send SNMP traps.
Step 2	Switch#**show env alarm-contact** **Example:** The output of the **show env alarm-contact** command is from the Cisco IE3500 Rugged Series Switch. `Switch#show env alarm-contact` `ALARM CONTACT 1` `Status: not asserted` `Description: door sensor` `Severity: major` `Trigger: closed` `ALARM CONTACT 2` `Status: not asserted` `Description: external alarm contact 2` `Severity: minor` `Trigger: closed`	(Optional) Verifies the configured alarm contacts.
Step 3	Switch(config)#**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Monitor and maintain switch alarms status

Use the show commands as required to display the switch alarms status.

- Use the **show alarm profile** *[name]* to display all alarm profiles in the system or a specified profile.
- Use the **show alarm settings** to display all global alarm settings on the switch.
  - The given output of the **show alarm settings** command is from the Cisco IE3500 Rugged Series Switch.

```
Switch#show alarm settings
Alarm relay mode: De-energized
Power Supply
         Alarm                    Enabled
         Relay
         Notifies                 Disabled
         Syslog                   Enabled
Temperature-Primary
         Alarm                    Enabled
         Thresholds               MAX: 95C                MIN:  -20C
         Relay                    MAJ
         Notifies                 Enabled
         Syslog                   Enabled
Temperature-Secondary
         Alarm                    Disabled
         Threshold
         Relay
         Notifies                 Disabled
```

```
         Syslog                  Disabled
SD-Card
         Alarm                   Disabled
         Relay
         Notifies                Disabled
         Syslog                  Enabled
Input-Alarm 1
         Alarm                   Enabled
         Relay
         Notifies                Disabled
         Syslog                  Enabled
Input-Alarm 2
         Alarm                   Enabled
         Relay
         Notifies                Disabled
         Syslog                  Enabled
```

- The given output of the **show alarm settings** command is from the Cisco IE3500H Heavy Duty Series Switch.

```
Switch#show alarm settings
Alarm relay mode: Positive
Temperature-Primary
         Alarm                   Enabled
         Thresholds              MAX: 80C                MIN:  0C
         Relay                   MAJ
         Notifies                Enabled
         Syslog                  Enabled
Temperature-Secondary
         Alarm                   Disabled
         Threshold
         Relay
         Notifies                Disabled
         Syslog                  Disabled
SD-Card
         Alarm                   Disabled
         Relay
         Notifies                Disabled
         Syslog                  Enabled
Input-Alarm 1
         Alarm                   Enabled
         Relay
         Notifies                Disabled
         Syslog                  Enabled
PTP
         Alarm                   Disabled
         Relay
         Notifies                Disabled
         Syslog                  Disabled
```

- Use the **show env** {**alarm-contact** | **all** | **power** | **temperature**} to display the status of environmental facilities on the switch.

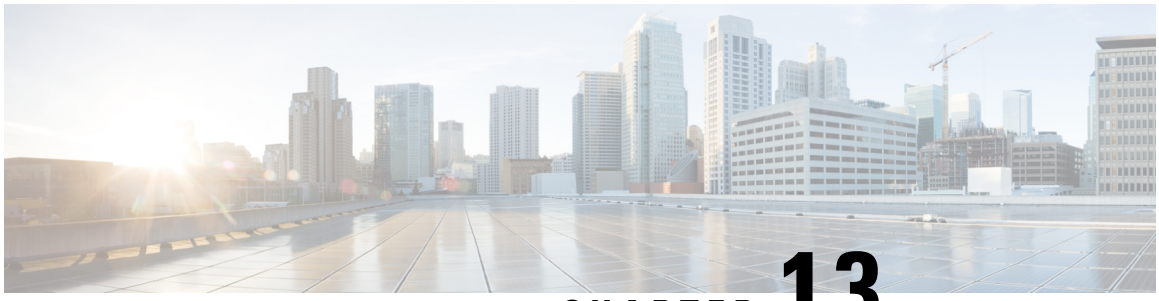
  The output of the **show env power** command is from the Cisco IE3500 Rugged Series Switch.

```
Switch#show env power
POWER SUPPLY A is DC OK
POWER SUPPLY B is DC FAULTY <--
Switch# show hard led
SWITCH: 1
SYSTEM: GREEN
ALARM : ALT_RED_BLACK <--
```

- Use the **show facility-alarm status** {**critical** | **info** | **major** | **minor**} to display generated alarms on the switch.

PART II

Layer2



CHAPTER 13

Configuring Spanning Tree Protocol

This chapter describes how to configure the Spanning Tree Protocol (STP) on port-based VLANs on the devices. The device can use either the per-VLAN spanning-tree plus (PVST+) protocol based on the IEEE 802.1D standard and Cisco proprietary extensions, or the rapid per-VLAN spanning-tree plus (rapid-PVST+) protocol based on the IEEE 802.1w standard.

Restrictions for Spanning Tree Protocol

- An attempt to configure a device as the root device fails if the value necessary to be the root device is less than 1.
- If your network consists of devices that support and do not support the extended system ID, it is unlikely that the device with the extended system ID support will become the root device. The extended system ID increases the device priority value every time the VLAN number is greater than the priority of the connected devices running older software.
- The root device for each spanning tree instance should be a backbone or distribution device. Do not configure an access device as the spanning tree primary root.

Information About Spanning Tree Protocol

The following sections provide information about spanning tree protocol:

Spanning Tree Protocol

Spanning Tree Protocol (STP) is a Layer 2 link management protocol that provides path redundancy while preventing loops in the network. For a Layer 2 Ethernet network to function properly, only one active path can exist between any two stations.

Multiple active paths among end stations cause loops in the network. If a loop exists in the network, end stations might receive duplicate messages. Devices might also learn end-station MAC addresses on multiple Layer 2 interfaces. These conditions result in an unstable network.

Spanning-tree operation is transparent to end stations, which cannot detect whether they are connected to a single LAN segment or a switched LAN of multiple segments.

The STP uses a spanning-tree algorithm to select one device of a redundantly connected network as the root of the spanning tree. The algorithm calculates the best loop-free path through a switched Layer 2 network by assigning a role to each port based on the role of the port in the active topology:

- Root—A forwarding port elected for the spanning-tree topology
- Designated—A forwarding port that is elected for every switched LAN segment
- Alternate—A blocked port providing an alternate path to the root bridge in the spanning tree
- Backup—A blocked port in a loopback configuration

The device that has *all* its ports as the designated role or as the backup role is the root device. The device that has at least *one* of its ports in the designated role is called the designated device.

Spanning tree forces redundant data paths into a standby (blocked) state. If a network segment in the spanning tree fails and a redundant path exists, the spanning-tree algorithm recalculates the spanning-tree topology and activates the standby path. Devices send and receive spanning-tree frames, called bridge protocol data units (BPDUs), at regular intervals. The devices do not forward these frames but use them to construct a loop-free path. BPDUs contain information about the sending device and its ports, including device and MAC addresses, device priority, port priority, and path cost. Spanning tree uses this information to elect the root device and root port for the switched network and the root port and designated port for each switched segment.

When two ports on a device are part of a loop, the spanning-tree and path cost settings control which port is put in the forwarding state and which is put in the blocking state. The spanning-tree port priority value represents the location of a port in the network topology and how well it is located to pass traffic. The path cost value represents the media speed.

Note The long path cost method is the default STP path cost method.

Note In addition to STP, the device uses keepalive messages to detect loops. By default, keepalive is enabled on Layer 2 ports. To disable keepalive, use the **no keepalive** command in interface configuration mode.

Spanning-Tree Topology and Bridge Protocol Data Units

The stable, active spanning-tree topology of a switched network is controlled by these elements:

- The unique bridge ID (device priority and MAC address) associated with each VLAN on each device.
- The spanning-tree path cost to the root device.
- The port identifier (port priority and MAC address) associated with each Layer 2 interface.

When the devices in a network are powered up, each functions as the root device. Each device sends a configuration BPDU through all its ports. The BPDUs communicate and compute the spanning-tree topology. Each configuration BPDU contains this information:

- The unique bridge ID of the device that the sending device identifies as the root device.
- The spanning-tree path cost to the root
- The bridge ID of the sending device
- Message age
- The identifier of the sending interface
- Values for the hello, forward delay, and max-age protocol timers

When a device receives a configuration BPDU that contains *superior* information (lower bridge ID, lower path cost, and so forth), it stores the information for that port. If this BPDU is received on the root port of the device, the device also forwards it with an updated message to all attached LANs for which it is the designated device.

If a device receives a configuration BPDU that contains *inferior* information to that currently stored for that port, it discards the BPDU. If the device is a designated device for the LAN from which the inferior BPDU was received, it sends that LAN a BPDU containing the up-to-date information stored for that port. In this way, inferior information is discarded, and superior information is propagated on the network.

A BPDU exchange results in these actions:

- One device in the network is elected as the root switch (the logical center of the spanning-tree topology in a switched network). See the figure following the bullets.

  For each VLAN, the device with the highest device priority (the lowest numerical priority value) is elected as the root switch. If all devices are configured with the default priority (32768), the devices with the lowest MAC address in the VLAN becomes the root device. The device priority value occupies the most significant bits of the bridge ID, .
- A root port is selected for each device (except the root switch). This port provides the best path (lowest cost) when the device forwards packets to the root switch.
- The shortest distance to the root switch is calculated for each device based on the path cost.
- A designated device for each LAN segment is selected. The designated device incurs the lowest path cost when forwarding packets from that LAN to the root switch. The port through which the designated device is attached to the LAN is called the designated port.

All paths that are not needed to reach the root switch from anywhere in the switched network are placed in the spanning-tree blocking mode.

Bridge ID, Device Priority, and Extended System ID

The IEEE 802.1D standard requires that each device has a unique bridge identifier (bridge ID), which controls the selection of the root switch. Because each VLAN is considered as a different *logical bridge* with PVST+ and Rapid PVST+, the same device must have a different bridge ID for each configured VLAN. Each VLAN on the device has a unique 8-byte bridge ID. The 2 most-significant bytes are used for the device priority, and the remaining 6 bytes are derived from the device MAC address.

The 2 bytes previously used for the device priority are reallocated into a 4-bit priority value and a 12-bit extended system ID value equal to the VLAN ID.

Table 14: Device Priority Value and Extended System ID

Priority Value				Extended System ID (Set Equal to the VLAN ID)											
Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

Spanning tree uses the extended system ID, the device priority, and the allocated spanning-tree MAC address to make the bridge ID unique for each VLAN.

Support for the extended system ID affects how you manually configure the root switch, the secondary root switch, and the switch priority of a VLAN. For example, when you change the switch priority value, you change the probability that the switch will be elected as the root switch. Configuring a higher value decreases the probability; a lower value increases the probability.

Port Priority Versus Path Cost

If a loop occurs, spanning tree uses port priority when selecting an interface to put into the forwarding state. You can assign higher priority values (lower numerical values) to interfaces that you want selected first and lower priority values (higher numerical values) that you want selected last. If all interfaces have the same priority value, spanning tree puts the interface with the lowest interface number in the forwarding state and blocks the other interfaces.

The spanning-tree path cost default value is derived from the media speed of an interface. If a loop occurs, spanning tree uses cost when selecting an interface to put in the forwarding state. You can assign lower cost values to interfaces that you want selected first and higher cost values that you want selected last. If all interfaces have the same cost value, spanning tree puts the interface with the lowest interface number in the forwarding state and blocks the other interfaces.

Spanning-Tree Interface States

Propagation delays can occur when protocol information passes through a switched LAN. As a result, topology changes can take place at different times and at different places in a switched network. When an interface transitions directly from nonparticipation in the spanning-tree topology to the forwarding state, it can create temporary data loops. Interfaces must wait for new topology information to propagate through the switched LAN before starting to forward frames. They must allow the frame lifetime to expire for forwarded frames that have used the old topology.

Each Layer 2 interface on a device using spanning tree exists in one of these states:

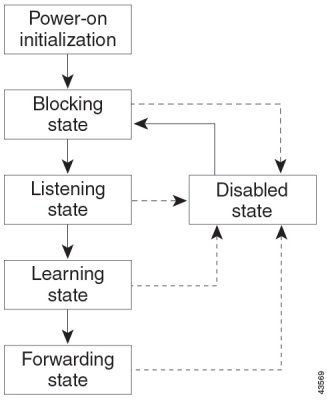
- Blocking—The interface does not participate in frame forwarding.
- Listening—The first transitional state after the blocking state when the spanning tree decides that the interface should participate in frame forwarding.
- Learning—The interface prepares to participate in frame forwarding.
- Forwarding—The interface forwards frames.
- Disabled—The interface is not participating in spanning tree because of a shutdown port, no link on the port, or no spanning-tree instance running on the port.

An interface moves through these states:

- From initialization to blocking
- From blocking to listening or to disabled
- From listening to learning or to disabled
- From learning to forwarding or to disabled
- From forwarding to disabled

Figure 2: Spanning-Tree Interface States

An interface moves through the states.



When you power up the device, spanning tree is enabled by default, and every interface in the device, VLAN, or network goes through the blocking state and the transitory states of listening and learning. Spanning tree stabilizes each interface at the forwarding or blocking state.

When the spanning-tree algorithm places a Layer 2 interface in the forwarding state, this process occurs:

1. The interface is in the listening state while spanning tree waits for protocol information to move the interface to the blocking state.
2. While spanning tree waits for the forward-delay timer to expire, it moves the interface to the learning state and resets the forward-delay timer.
3. In the learning state, the interface continues to block frame forwarding as the device learns end-station location information for the forwarding database.
4. When the forward-delay timer expires, spanning tree moves the interface to the forwarding state, where both learning and frame forwarding are enabled.

Blocking State

A Layer 2 interface in the blocking state does not participate in frame forwarding. After initialization, a BPDU is sent to each device interface. A device initially functions as the root until it exchanges BPDUs with other devices. This exchange establishes which device in the network is the root or root device. If there is only one device in the network, no exchange occurs, the forward-delay timer expires, and the interface moves to the listening state. An interface always enters the blocking state after device initialization.

An interface in the blocking state performs these functions:

- Discards frames received on the interface

- Discards frames that are switched from another interface for forwarding
- Does not learn addresses
- Receives BPDUs

Listening State

The listening state is the first state a Layer 2 interface enters after the blocking state. The interface enters this state when the spanning tree decides that the interface should participate in frame forwarding.

An interface in the listening state performs these functions:

- Discards frames received on the interface
- Discards frames that are switched from another interface for forwarding
- Does not learn addresses
- Receives BPDUs

Learning State

A Layer 2 interface in the learning state prepares to participate in frame forwarding. The interface enters the learning state from the listening state.

An interface in the learning state performs these functions:

- Discards frames received on the interface
- Discards frames that are switched from another interface for forwarding
- Learns addresses
- Receives BPDUs

Forwarding State

A Layer 2 interface in the forwarding state forwards frames. The interface enters the forwarding state from the learning state.

An interface in the forwarding state performs these functions:

- Receives and forwards frames that are received on the interface.
- Forwards frames that are switched from another interface
- Learns addresses
- Receives BPDUs

Disabled State

A Layer 2 interface in the disabled state does not participate in frame forwarding or in the spanning tree. An interface in the disabled state is nonoperational.

A disabled interface performs these functions:

- Discards frames received on the interface

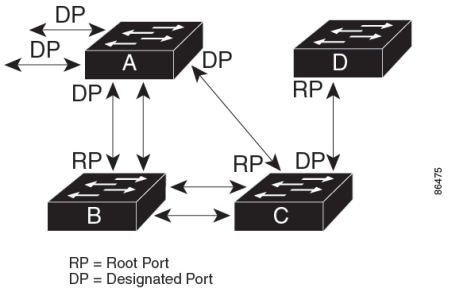
- Discards frames that are switched from another interface for forwarding
- Does not learn addresses
- Does not receive BPDUs

How a Device or Port Becomes the Root Device or Root Port

If all devices in a network are enabled with default spanning-tree settings, the device with the lowest MAC address becomes the root device.

Figure 3: Spanning-Tree Topology

Switch A is elected as the root device because the device priority of all the devices is set to the default (32768) and Switch A has the lowest MAC address. However, because of traffic patterns, number of forwarding interfaces, or link types, Switch A might not be the ideal root device. By increasing the priority (lowering the numerical value) of the ideal device so that it becomes the root device, you force a spanning-tree recalculation to form a new topology with the ideal device as the root.



When the spanning-tree topology is calculated based on default parameters, the path between source and destination end stations in a switched network might not be ideal. For instance, connecting higher-speed links to an interface that has a higher number than the root port can cause a root-port change. The goal is to make the fastest link the root port.

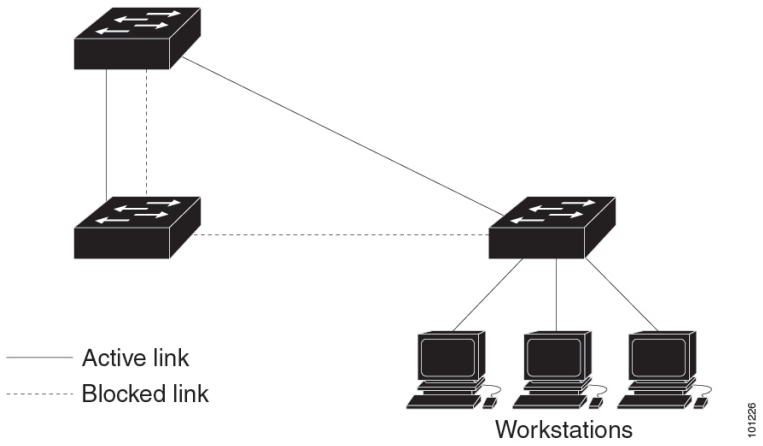
For example, assume that one port on Switch B is a Gigabit Ethernet link and that another port on Switch B (a 10/100 link) is the root port. Network traffic might be more efficient over the Gigabit Ethernet link. By changing the spanning-tree port priority on the Gigabit Ethernet port to a higher priority (lower numerical value) than the root port, the Gigabit Ethernet port becomes the new root port.

Spanning Tree and Redundant Connectivity

You can create a redundant backbone with spanning tree by connecting two switch interfaces to another device or to two different devices. Spanning tree automatically disables one interface but enables it if the other one fails. If one link is high-speed and the other is low-speed, the low-speed link is always disabled. If the speeds are the same, the port priority and port ID are added together, and spanning tree disables the link with the highest value.

Figure 4: Spanning Tree and Redundant Connectivity, on page 116 shows redundant connectivity on a spanning tree topology.

Figure 4: Spanning Tree and Redundant Connectivity



You can also create redundant links between devices by using EtherChannel groups.

Spanning-Tree Address Management

IEEE 802.1D specifies 17 multicast addresses, ranging from 0x00180C2000000 to 0x0180C2000010, to be used by different bridge protocols. These addresses are static addresses that cannot be removed.

If spanning tree is enabled, the CPU on the switch receives packets that are destined for 0x0180C2000000 and 0x0180C2000010. If spanning tree is disabled, the switch forwards those packets as unknown multicast addresses.

Accelerated Aging to Retain Connectivity

The default for aging dynamic addresses is 5 minutes, the default setting of the **mac address-table aging-time** global configuration command. However, a spanning-tree reconfiguration can cause many station locations to change. Because these stations could be unreachable for 5 minutes or more during a reconfiguration, the address-aging time is accelerated so that station addresses can be dropped from the address table and then relearned. The accelerated aging is the same as the forward-delay parameter value (**spanning-tree vlan** *vlan-id* **forward-time** *seconds* global configuration command) when the spanning tree reconfigures.

Because each VLAN is a separate spanning-tree instance, the switch accelerates aging on a per-VLAN basis. A spanning-tree reconfiguration on one VLAN can cause the dynamic addresses that are learned on that VLAN to be subject to accelerated aging. Dynamic addresses on other VLANs can be unaffected and remain subject to the aging interval entered for the switch.

Spanning-Tree Modes and Protocols

The device supports these spanning-tree modes and protocols:

- PVST+—This spanning-tree mode is based on the IEEE 802.1D standard and Cisco proprietary extensions. The PVST+ runs on each VLAN on the device up to the maximum supported, ensuring that each has a loop-free path through the network.

  The PVST+ provides Layer 2 load-balancing for the VLAN on which it runs. You can create different logical topologies by using the VLANs on your network to ensure that all of your links are used but that no one link is oversubscribed. Each instance of PVST+ on a VLAN has a single root switch. This root switch propagates the spanning-tree information that is associated with that VLAN to all other devices

in the network. Because each device has the same information about the network, this process ensures that the network topology is maintained.

- Rapid PVST+—This spanning-tree mode is the same as PVST+ except that is uses a rapid convergence based on the IEEE 802.1w standard. Rapid PVST+ is the default STP mode on your device. This spanning-tree mode is the same as PVST+ except that is uses a rapid convergence based on the IEEE 802.1w standard. To provide rapid convergence, the Rapid PVST+ immediately deletes dynamically learned MAC address entries on a per-port basis upon receiving a topology change. By contrast, PVST+ uses a short aging time for dynamically learned MAC address entries.

  Rapid PVST+ uses the same configuration as PVST+ (except where noted), and the device needs only minimal extra configuration. The benefit of Rapid PVST+ is that you can migrate a large PVST+ install base to Rapid PVST+ without having to learn the complexities of the Multiple Spanning Tree Protocol (MSTP) configuration and without having to reprovision your network. In Rapid PVST+ mode, each VLAN runs its own spanning-tree instance up to the maximum supported.

- MSTP—This spanning-tree mode is based on the IEEE 802.1s standard. You can map multiple VLANs to the same spanning-tree instance, which reduces the number of spanning-tree instances that are required to support many VLANs. The MSTP runs on top of the RSTP (based on IEEE 802.1w), which provides for rapid convergence of the spanning tree by eliminating the forward delay and by quickly transitioning root ports and designated ports to the forwarding state.

Supported Spanning-Tree Instances

In PVST+ or Rapid PVST+ mode, the device supports up to 128 spanning-tree instances.

In MSTP mode, the device supports up to 64 MST instances. The number of VLANs that can be mapped to a particular MST instance is 512.

Spanning-Tree Interoperability and Backward Compatibility

In a mixed MSTP and PVST+ network, the common spanning-tree (CST) root must be inside the MST backbone, and a PVST+ device cannot connect to multiple MST regions.

When a network contains devices running Rapid PVST+ and devices running PVST+, we recommend that the Rapid PVST+ devices and PVST+ devices be configured for different spanning-tree instances. In the Rapid PVST+ spanning-tree instances, the root switch must be a Rapid PVST+ device. In the PVST+ instances, the root switch must be a PVST+ device. The PVST+ devices should be at the edge of the network.

Table 15: PVST+, MSTP, and Rapid-PVST+ Interoperability and Compatibility

	PVST+	**MSTP**	**Rapid PVST+**
PVST+	Yes	Yes (with restrictions)	Yes (reverts to PVST+)
MSTP	Yes (with restrictions)	Yes	Yes (reverts to PVST+)
Rapid PVST+	Yes (reverts to PVST+)	Yes (reverts to PVST+)	Yes

Spanning Tree Protocols and IEEE 802.1Q Trunks

The IEEE 802.1Q standard for VLAN trunks imposes some limitations on the spanning-tree strategy for a network. The standard requires only one spanning-tree instance for *all* VLANs allowed on the trunks. However,

in a network of Cisco devices that are connected through IEEE 802.1Q trunks, the devices maintain one spanning-tree instance for *each* VLAN allowed on the trunks.

When you connect a Cisco device to a non-Cisco device through an IEEE 802.1Q trunk, the Cisco device uses PVST+ to provide spanning-tree interoperability. If Rapid PVST+ is enabled, the device uses it instead of PVST+. The device combines the spanning-tree instance of the IEEE 802.1Q VLAN of the trunk with the spanning-tree instance of the non-Cisco IEEE 802.1Q device.

However, all PVST+ or Rapid PVST+ information is maintained by Cisco devices that are separated by a cloud of non-Cisco IEEE 802.1Q devices. The non-Cisco IEEE 802.1Q cloud separating the Cisco devices is treated as a single trunk link between the devices.

Rapid PVST+ is automatically enabled on IEEE 802.1Q trunks, and no user configuration is required. The external spanning-tree behavior on access ports and Inter-Switch Link (ISL) trunk ports is not affected by PVST+.

Default Spanning-Tree Configuration

Table 16: Default Spanning-Tree Configuration

Feature	Default Setting
Enable state	Enabled on VLAN 1.
Spanning-tree mode	Rapid PVST+ (PVST+ and MSTP are disabled.)
Device priority	32768
Spanning-tree port priority (configurable on a per-interface basis)	128
Spanning-tree port cost (configurable on a per-interface basis)	10 Mbps: 2000000 100 Mbps: 200000 1 Gbps: 20000 10 Gbps: 2000 40 Gbps: 500 100 Gbps: 200 1 Tbps: 20 10 Tbps: 2
Spanning-tree VLAN port priority (configurable on a per-VLAN basis)	128

Feature	Default Setting
Spanning-tree VLAN port cost (configurable on a per-VLAN basis)	10 Mbps: 2000000 100 Mbps: 200000 1 Gbps: 20000 10 Gbps: 2000 40 Gbps: 500 100 Gbps: 200 1 Tbps: 20 10 Tbps: 2
Spanning-tree timers	Hello time: 2 seconds Forward-delay time: 15 seconds Maximum-aging time: 20 seconds Transmit hold count: 6 BPDUs

How to Configure Spanning Tree Protocol

The following sections provide information about configuring spanning tree protocol:

Changing the Spanning-Tree Mode

The switch supports three spanning-tree modes: per-VLAN spanning tree plus (PVST+), Rapid PVST+, or Multiple Spanning Tree Protocol (MSTP). By default, the device runs the Rapid PVST+ protocol.

If you want to enable a mode that is different from the default mode, this procedure is required.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mode** {**pvst** | **mst** | **rapid-pvst**} **Example:** `Device(config)# spanning-tree mode pvst`	Configures a spanning-tree mode. • Select **pvst** to enable PVST+. • Select **mst** to enable MSTP.

	Command or Action	Purpose
		• Select **rapid-pvst** to enable rapid PVST+.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet1/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports, VLANs, and port channels. The VLAN ID range is 1 to 4094. The port-channel range is 1 to 48.
Step 5	**spanning-tree link-type point-to-point** **Example:** `Device(config-if)# spanning-tree link-type point-to-point`	Specifies that the link type for this port is point-to-point. If you connect this port (local port) to a remote port through a point-to-point link and the local port becomes a designated port, the device negotiates with the remote port and rapidly changes the local port to the forwarding state.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 7	**clear spanning-tree detected-protocols** **Example:** `Device# clear spanning-tree detected-protocols`	If any port on the device is connected to a port on a legacy IEEE 802.1D device, this command restarts the protocol migration process on the entire device. This step is optional if the designated device detects that this device is running rapid PVST+.

(Optional) Disabling Spanning Tree

Spanning tree is enabled by default on VLAN 1 and on all newly created VLANs up to the spanning-tree limit. Disable spanning tree only if you are sure that there are no loops in the network topology.

Caution When spanning tree is disabled and loops are present in the topology, excessive traffic and indefinite packet duplication can drastically reduce network performance.

To disable spanning tree, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no spanning-tree vlan** *vlan-id* **Example:** `Device(config)# no spanning-tree vlan 300`	For *vlan-id*, the range is 1 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Root Device

To configure a device as the root for the specified VLAN, use the **spanning-tree vlan** *vlan-id* **root** global configuration command to modify the device priority from the default value (32768) to a significantly lower value. When you enter this command, the software checks the switch priority of the root switches for each VLAN. Because of the extended system ID support, the switch sets its own priority for the specified VLAN to 24576 if this value causes this switch to become the root for the specified VLAN.

Use the **diameter** keyword to specify the Layer 2 network diameter (that is, the maximum number of device hops between any two end stations in the Layer 2 network). When you specify the network diameter, the device automatically sets an optimal hello time, forward-delay time, and maximum-age time for a network of that diameter, which can significantly reduce the convergence time. You can use the **hello** keyword to override the automatically calculated hello time.

To configure the root device, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **root primary** [**diameter** *net-diameter*] **Example:** `Device(config)# spanning-tree vlan 20-24 root primary diameter 4`	Configures a device to become the root for the specified VLAN. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen,

	Command or Action	Purpose
		or a series of VLANs separated by a comma. The range is 1 to 4094. • (Optional) For **diameter** *net-diameter*, specify the maximum number of devices between any two end stations. The range is 2 to 7.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

What to do next

After configuring the switch as the root switch, we recommend that you avoid manually configuring the hello time, forward-delay time, and maximum-age time through the **spanning-tree vlan** *vlan-id* **hello-time**, **spanning-tree vlan** *vlan-id* **forward-time**, and the **spanning-tree vlan** *vlan-id* **max-age** global configuration commands.

(Optional) Configuring a Secondary Root Device

When you configure a switch as the secondary root, the switch priority is modified from the default value (32768) to 28672. With this priority, the switch is likely to become the root switch for the specified VLAN if the primary root switch fails. This is assuming that the other network switches use the default switch priority of 32768, and therefore, are unlikely to become the root switch.

You can execute this command on more than one switch to configure multiple backup root switches. Use the same network diameter and hello-time values that you used when you configured the primary root switch with the **spanning-tree vlan** *vlan-id* **root primary** global configuration command.

To configure a secondary root device, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **root secondary** [**diameter** *net-diameter*] **Example:**	Configures a device to become the secondary root for the specified VLAN. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a

	Command or Action	Purpose
	`Device(config)# spanning-tree vlan 20-24 root secondary diameter 4`	range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • (Optional) For **diameter** *net-diameter*, specify the maximum number of devices between any two end stations. The range is 2 to 7. Use the same network diameter value that you used when configuring the primary root switch.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring Port Priority

To configure port priority, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports and port-channel logical interfaces (**port-channel** *port-channel-number*).
Step 4	**spanning-tree port-priority** *priority* **Example:** `Device(config-if)# spanning-tree port-priority 0`	Configures the port priority for an interface. For *priority*, the range is 0 to 240, in increments of 16; the default is 128. Valid values are 0, 16, 32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, 224, and 240. All other values are rejected. The lower the number, the higher the priority.
Step 5	**spanning-tree vlan** *vlan-id* **port-priority** *priority*	Configures the port priority for a VLAN.

	Command or Action	Purpose
	Example: `Device(config-if)# spanning-tree vlan 20-25 port-priority 0`	• For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *priority*, the range is 0 to 240, in increments of 16; the default is 128. Valid values are 0, 16, 32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, 224, and 240. All other values are rejected. The lower the number, the higher the priority.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

(Optional) Configuring Path Cost

To configure path cost, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports and port-channel logical interfaces (**port-channel** *port-channel-number*).
Step 4	**spanning-tree cost** *cost* **Example:** `Device(config-if)# spanning-tree cost 250`	Configures the cost for an interface. If a loop occurs, spanning tree uses the path cost when selecting an interface to place into the forwarding state. A lower path cost represents higher-speed transmission. For *cost*, the range is 1 to 200000000; the default value is derived from the media speed of the interface.

	Command or Action	Purpose
Step 5	**spanning-tree vlan** *vlan-id* **cost** *cost* **Example:** `Device(config-if)# spanning-tree vlan 10,12-15,20 cost 300`	Configures the cost for a VLAN. If a loop occurs, spanning tree uses the path cost when selecting an interface to place into the forwarding state. A lower path cost represents higher-speed transmission. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *cost*, the range is 1 to 200000000; the default value is derived from the media speed of the interface.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

The **show spanning-tree interface** *interface-id* privileged EXEC command displays information only for ports that are in a link-up operative state. Otherwise, you can use the **show running-config** privileged EXEC command to confirm the configuration.

(Optional) Configuring the Device Priority of a VLAN

You can configure the switch priority and make it more likely that a standalone switch will be chosen as the root switch.

Note Exercise care when using this command. For most situations, we recommend that you use the **spanning-tree vlan** *vlan-id* **root primary** and the **spanning-tree vlan** *vlan-id* **root secondary** global configuration commands to modify the switch priority.

To configure device priority of a VLAN, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**spanning-tree vlan** *vlan-id* **priority** *priority* **Example:** `Device(config)# spanning-tree vlan 20 priority 8192`	Configures the device priority of a VLAN. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *priority*, the range is 0 to 61440 in increments of 4096; the default is 32768. The lower the number, the more likely the switch will be chosen as the root switch. Valid priority values are 4096, 8192, 12288, 16384, 20480, 24576, 28672, 32768, 36864, 40960, 45056, 49152, 53248, 57344, and 61440. All other values are rejected.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Hello Time

The hello time is the time interval between configuration messages that are generated and sent by the root switch.

To configure the hello time, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**spanning-tree vlan** *vlan-id* **hello-time** *seconds* **Example:** `Device(config)# spanning-tree vlan 20-24 hello-time 3`	Configures the hello time of a VLAN. The hello time is the time interval between configuration messages that are generated and sent by the root switch. These messages mean that the switch is alive. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen,

	Command or Action	Purpose
		or a series of VLANs separated by a comma. The range is 1 to 4094. • For *seconds*, the range is 1 to 10; the default is 2.
Step 3	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Forwarding-Delay Time for a VLAN

To configure the forwarding-delay time for a VLAN, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **forward-time** *seconds* **Example:** `Device(config)# spanning-tree vlan 20,25 forward-time 18`	Configures the forward time of a VLAN. The forwarding delay is the number of seconds an interface waits before changing from its spanning-tree learning and listening states to the forwarding state. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *seconds*, the range is 4 to 30; the default is 15.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Maximum-Aging Time for a VLAN

To configure the maximum-aging time for a VLAN, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **max-age** *seconds* **Example:** `Device(config)# spanning-tree vlan 20 max-age 30`	Configures the maximum-aging time of a VLAN. The maximum-aging time is the number of seconds a switch waits without receiving spanning-tree configuration messages before attempting a reconfiguration. • For *vlan-id*, you can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094. • For *seconds*, the range is 6 to 40; the default is 20.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Transmit Hold-Count

You can configure the BPDU burst size by changing the transmit hold count value.

Note Changing this parameter to a higher value can have a significant impact on CPU utilization, especially in Rapid PVST+ mode. Lowering this value can slow down convergence in certain scenarios. We recommend that you maintain the default setting.

To configure the transmit hold-count, perform this procedure:

Procedure

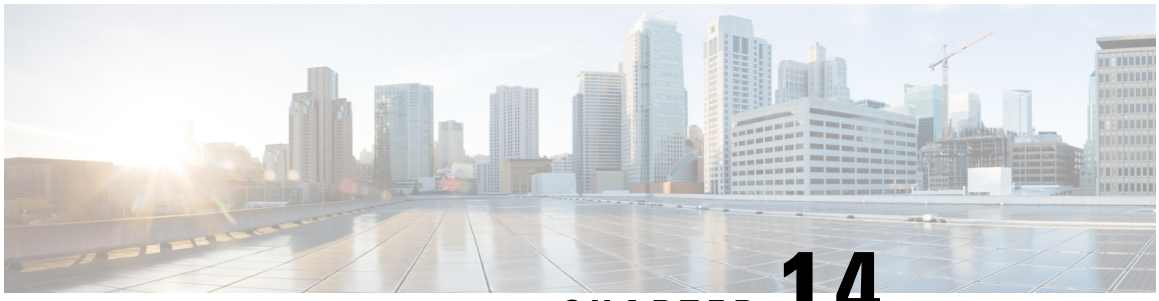
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree transmit hold-count** *value* **Example:** `Device(config)# spanning-tree transmit hold-count 6`	Configures the number of BPDUs that can be sent before pausing for 1 second. For *value*, the range is 1 to 20; the default is 6.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Monitoring Spanning Tree Protocol Configuration Status

Table 17: Commands for Displaying STP Configuration Status

show spanning-tree active	Displays STP configuration information on active interfaces only.
show spanning-tree detail	Displays a detailed summary of interface information.
show spanning-tree vlan *vlan-id*	Displays STP configuration information for the specified VLAN.
show spanning-tree interface *interface-id*	Displays STP configuration information for the specified interface.
show spanning-tree interface *interface-id* **portfast**	Displays STP portfast information for the specified interface.
show spanning-tree summary [**totals**]	Displays a summary of interface states or displays the total lines of the STP state section.

To clear STP counters, use the **clear spanning-tree** [**interface** i*nterface-id*] privileged EXEC command.



CHAPTER 14

Configuring Loop Detection Guard

Restrictions for Loop Detection Guard

Loop detection guard can be configured only on Layer 2 physical interfaces. Layer 3 ports and virtual interfaces, such as port channels, switch virtual interfaces (SVIs), and tunnels, are not supported.

Information About Loop Detection Guard

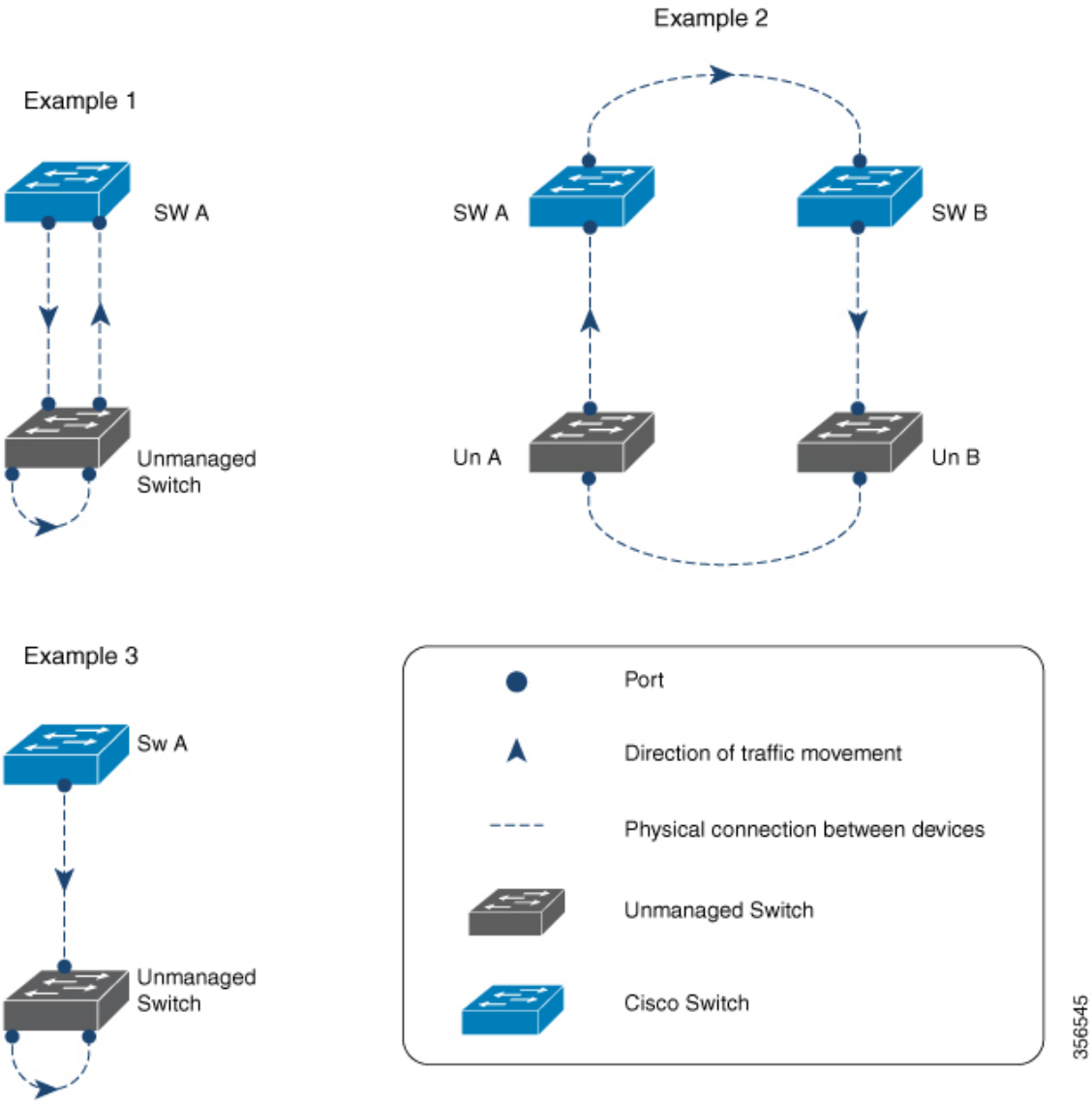
A computer network can experience a network loop where there is more than one Layer 2 path between two endpoints. This is possible when there are multiple connections between two switches in a network or two ports on the same switch are connected to each other. The following figure shows a few examples of a network loop:

Example 1: Switch SW A, which is within the network, is sending traffic to an unmanaged switch on one port and receiving traffic from the same unmanaged switch, on another port. On the unmanaged switch, the port receiving traffic is connected to the port sending traffic back to the SW A in the network, resulting in a network loop.

Example 2: This example shows a network loop involving four switches, two within the network (SW A and SW B) and two unmanaged switches (Un A and Un B). Traffic is moving in the following direction SW A to SW B to Un B to Un A and back to SW A, resulting in a network loop.

Example 3: Two ports on the unmanaged switch are connected to each other, resulting in a network loop.

Figure 5: Examples of Network Loop Between Managed and Unmanaged Switches



While Spanning Tree Protocol (STP) is normally the protocol that is configured for this purpose (to prevent network loops), loop detection guard is suited to situations where there may be unmanaged switches in a network that do not understand STP, or where STP is not configured on the network.

Loop detection guard is enabled at the interface level. To detect loops, the system sends loop-detect frames from the interface, at preconfigured intervals. When a loop is detected, the configured action is taken.

Loop detection guard is disabled by default. When you enable the feature, you can configure one of these actions:

- Error-disable the port sending traffic.
- Error-disable the port receiving traffic (default).
- Display an error message and not disable any port.

When a port is error-disabled, no traffic is sent or received on that port.

Interaction of Loop Detection Guard with Other Features

The following sections provide information about how loop detection guard interacts with other feature:

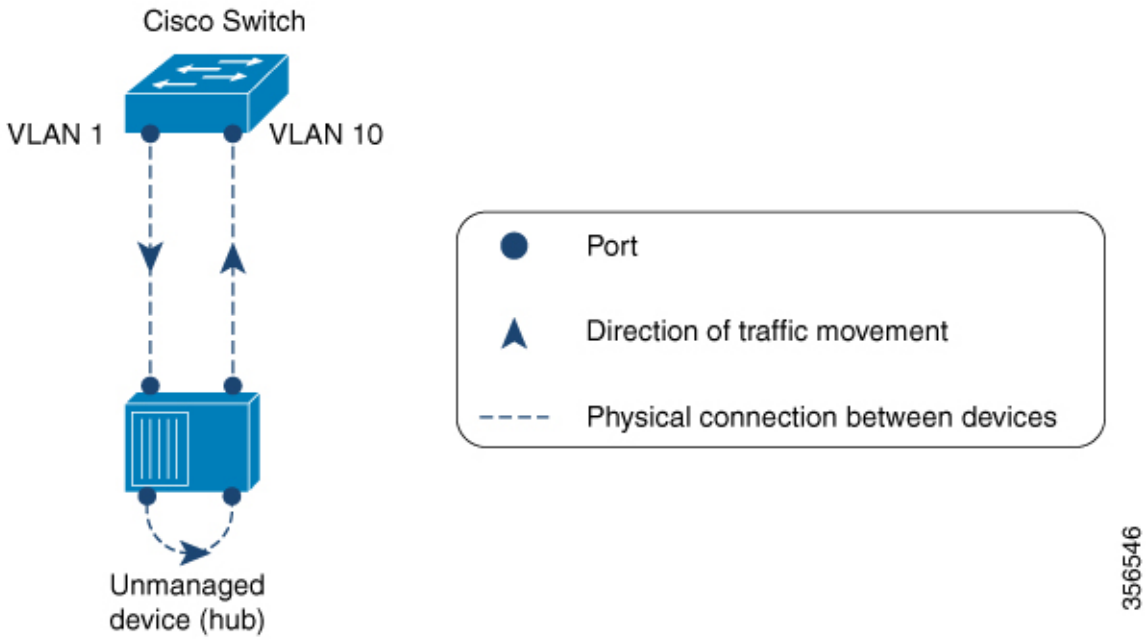
Spanning Tree Protocol and Loop Detection Guard

When both loop detection guard and STP are enabled on a device, STP takes over monitoring the network for loops. In this case loop-detect packets are neither received nor processed in the network.

VLANs and Loop Detection Guard

We do not recommend configuring this feature on a switch that is connected to a hub for these reasons: The hub floods traffic to all of its interfaces. If the switch in your network is receiving traffic from the same hub, but on a port in a different VLAN, you may be inadvertently error-disabling those destination ports. The figure below illustrates such a situation. The port in VLAN 1 is sending traffic to the hub. The switch is also receiving traffic from the same hub, but on a port in a different VLAN, that is, VLAN 10. If you configure loop detection guard (and you have configured the default action of error-disabling the destination port), then the port in VLAN 10 is blocked. Configuring the option to display a message (instead of error-disabling a port) is not recommended either, because the system displays as many messages as the number of interfaces configured in the hub, resulting in a CPU overload.

Figure 6: A Switch Connected to an Unmanged Network Hub



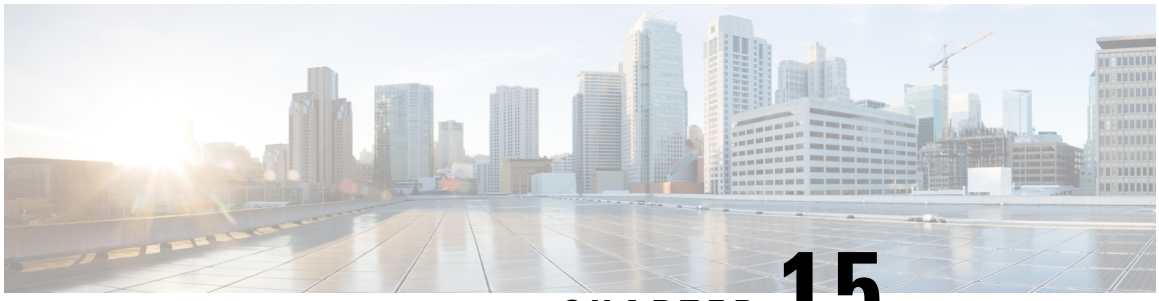
Enabling Loop Detection Guard and Error-Disabling the Required Port

The feature is disabled by default. Complete the following steps to enable loop detection guard and configure the action that you want the system to take when a loop is detected:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface***interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1` `Device(config-if)#`	Enters interface configuration mode. Specify only a physical interface to configure loop detection guard on the device. Layer 3 ports and virtual interfaces like PortChannels, switch virtual interfaces (SVIs), and tunnels are not supported.
Step 4	[**no**] **loopdetect** **Example:** `Device(config-if)# loopdetect`	Enables loop detection guard on the device. Loopdetect frames are sent from the configured interface. Use the **loopdetect** command without any keyword to enable loop detection guard. Use the **no** form of this command to disable this feature. **Note** You can enable the feature on trunk ports, but a warning message is displayed, for the following reason: A trunk port carries traffic for several VLANs, simultaneously. A loop that is detected in one VLAN can result in the error-disabling of all VLAN traffic that is associated with the trunk port.
Step 5	[**no**] **loopdetect** {*time* | **action syslog** | **source-port**} **Example:** `Device(config-if)# loopdetect 7`	Specifies the frequency at which loop-detect frames are sent and the action the system takes when a loop is detected. If you do not specify an action, the destination port is error-disabled by default. You can configure the following:

	Command or Action	Purpose
		• *time*—Time interval to send loop-detect frame, in seconds. The range is from 1 to 10. The default is 5. • **action syslog**—Displays a system message and does not error-disable any port. If you use the **no** form of this command, the system reverts to the last configured option. • **source-port**—Error-disables the source port. If you use the **no** form of this command, the destination port is error-disabled. In the example configuration on the left (`Device(config-if)# loopdetect 7`), the interface is configured to send loop-detect frames every 7 seconds, and to error-disable the destination port if a loop is detected (The default applies, because neither the **action syslog** option nor the **source-port** option has been configured).
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 7	**show loopdetect** **Example:** `Device# show loopdetect`	Displays all the interfaces where loop detection guard is enabled, the frequency at which loop-detect packets are sent, and the status of the physical interface.



CHAPTER 15

Configuring Multiple Spanning-Tree Protocol

Prerequisites for Multiple Spanning Tree Protocol

- For two or more devices to be in the same multiple spanning tree (MST) region, they must have the same VLAN-to-instance map, the same configuration revision number, and the same name.
- For load-balancing across redundant paths in the network to work, all VLAN-to-instance mapping assignments must match; otherwise, all traffic flows on a single link.
- For load-balancing between a per-VLAN spanning tree plus (PVST+) and an MST cloud or between a rapid-PVST+ and an MST cloud to work, all MST boundary ports must be forwarding. MST boundary ports are forwarding when the root of the internal spanning tree (IST) of the MST cloud is the root of the common spanning tree (CST). If the MST cloud consists of multiple MST regions, one of the MST regions must contain the CST root, and all of the other MST regions must have a better path to the root contained within the MST cloud than a path through the PVST+ or rapid-PVST+ cloud. You might have to manually configure the devices in the clouds.

Restrictions for Multiple Spanning-Tree Protocol

- PVST+, Rapid PVST+, and MSTP are supported, but only one version can be active at any time. (For example, all VLANs run PVST+, all VLANs run Rapid PVST+, or all VLANs run MSTP.)
- VLAN Trunking Protocol (VTP) propagation of the MST configuration is not supported. However, you can manually configure the MST configuration (region name, revision number, and VLAN-to-instance mapping) on each device within the MST region by using the command-line interface (CLI) or through the Simple Network Management Protocol (SNMP) support.
- Partitioning the network into a large number of regions is not recommended. However, if this situation is unavoidable, we recommend that you partition the switched LAN into smaller LANs interconnected by routers or non-Layer 2 devices.

- A region can have one member or multiple members with the same MST configuration; each member must be capable of processing rapid spanning tree protocol (RSTP) Bridge Protocol Data Units (BPDUs). There is no limit to the number of MST regions in a network, but each region can only support up to 65spanning-tree instances. You can assign a VLAN to only one spanning-tree instance at a time.

Information About Multiple Spanning Tree Protocol

The following sections provide information about Multiple Spanning-Tree Protocol (MSTP):

Multiple Spanning Tree Protocol Configuration

Multiple Spanning-Tree Protocol (MSTP), which uses Rapid Spanning-Tree Protocol (RSTP) for rapid convergence, enables multiple VLANs to be grouped into and mapped to the same spanning-tree instance, reducing the number of spanning-tree instances that are needed to support many VLANs. The MSTP provides for multiple forwarding paths for data traffic, enables load balancing, and reduces the number of spanning-tree instances that are required to support many VLANs. It improves the fault tolerance of the network because a failure in one instance (forwarding path) does not affect other instances (forwarding paths).

Note The multiple spanning-tree (MST) implementation is based on the IEEE 802.1s standard.

The most common initial deployment of MSTP is in the backbone and distribution layers of a Layer 2 switched network. This deployment provides the highly available network that is required in a service-provider environment.

When the device is in the MST mode, the RSTP, which is based on IEEE 802.1w, is automatically enabled. The RSTP provides rapid convergence of the spanning tree through explicit handshaking that eliminates the IEEE 802.1D forwarding delay and quickly transitions root ports and designated ports to the forwarding state.

Both MSTP and RSTP improve the spanning-tree operation and maintain backward compatibility with equipment that is based on the (original) IEEE 802.1D spanning tree, with existing Cisco-proprietary Multiple Instance STP (MISTP), and with existing Cisco PVST+ and rapid per-VLAN spanning-tree plus (Rapid PVST+).

In MSTP mode, a device supports up to 64 MST instances. The number of VLANs that can be mapped to a particular MST instance is 512.

Multiple Spanning Tree Protocol Configuration Guidelines

- When you enable MST by using the **spanning-tree mode mst** global configuration command, RSTP is automatically enabled.
- For configuration guidelines about UplinkFast and BackboneFast, see the relevant sections in the Related Topics section.
- When the device is in MST mode, it uses the long path-cost calculation method (32 bits) to compute the path cost values. With the long path-cost calculation method, the following path cost values are supported:

Speed	Path Cost Value
10 Mb/s	2,000,000
100 Mb/s	200,000
1 Gb/s	20,000
10 Gb/s	2,000
100 Gb/s	200

Root Switch Configuration

The switch maintains a spanning-tree instance for the group of VLANs mapped to it. A device ID, consisting of the switch priority and the switch MAC address, is associated with each instance. For a group of VLANs, the switch with the lowest device ID becomes the root switch.

When you configure a switch as the root, you modify the switch priority from the default value (32768) to a significantly lower value so that the switch becomes the root switch for the specified spanning-tree instance. When you enter this command, the switch checks the switch priorities of the root switches. Because of the extended system ID support, the switch sets its own priority for the specified instance to 24576 if this value will cause this switches to become the root for the specified spanning-tree instance.

If any root switch for the specified instance has a switch priority lower than 24576, the switch sets its own priority to 4096 less than the lowest switch priority. (4096 is the value of the least-significant bit of a 4-bit switch priority value. For more information, see Bridge ID, Switch Priority, and Extended System ID.

If your network consists of switches that support and do not support the extended system ID, it is unlikely that the switch with the extended system ID support will become the root switch. The extended system ID increases the switch priority value every time the VLAN number is greater than the priority of the connected switches running older software.

The root switch for each spanning-tree instance should be a backbone or distribution switch. Do not configure an access switch as the spanning-tree primary root.

Use the **diameter** keyword, which is available only for MST instance 0, to specify the Layer 2 network diameter (that is, the maximum number of switch hops between any two end stations in the Layer 2 network). When you specify the network diameter, the switch automatically sets an optimal hello time, forward-delay time, and maximum-age time for a network of that diameter, which can significantly reduce the convergence time. You can use the **hello** keyword to override the automatically calculated hello time.

Multiple Spanning-Tree Regions

For switches to participate in multiple spanning-tree (MST) instances, you must consistently configure the switches with the same MST configuration information. A collection of interconnected switches that have the same MST configuration comprises an MST region.

The MST configuration controls to which MST region each device belongs. The configuration includes the name of the region, the revision number, and the MST VLAN-to-instance assignment map. You configure the device for a region by specifying the MST region configuration on it. You can map VLANs to an MST instance, specify the region name, and set the revision number. For instructions and an example, select the "Specifying the MST Region Configuration and Enabling MSTP" link in Related Topics.

A region can have one or multiple members with the same MST configuration. Each member must be capable of processing RSTP bridge protocol data units (BPDUs). There is no limit to the number of MST regions in a network, but each region can support up to 64 spanning-tree instances. Instances can be identified by any number in the range from 0 to 4094. You can assign a VLAN to only one spanning-tree instance at a time.

Internal Spanning Tree, Common and Internal Spanning Tree, and Common Spanning Tree

Unlike PVST+ and Rapid PVST+ in which all the spanning-tree instances are independent, the MSTP establishes and maintains two types of spanning trees:

- An internal spanning tree (IST), which is the spanning tree that runs in an MST region.

  Within each MST region, the MSTP maintains multiple spanning-tree instances. Instance 0 is a special instance for a region, known as the internal spanning tree (IST). All other MST instances are numbered from 1 to 4094.

  The IST is the only spanning-tree instance that sends and receives BPDUs. All of the other spanning-tree instance information is contained in M-records, which are encapsulated within MSTP BPDUs. Because the MSTP BPDU carries information for all instances, the number of BPDUs that need to be processed to support multiple spanning-tree instances is significantly reduced.

  All MST instances within the same region share the same protocol timers, but each MST instance has its own topology parameters, such as root switch ID, root path cost, and so forth. By default, all VLANs are assigned to the IST.

  An MST instance is local to the region; for example, MST instance 1 in region A is independent of MST instance 1 in region B, even if regions A and B are interconnected.

- A common and internal spanning tree (CIST), which is a collection of the ISTs in each MST region, and the common spanning tree (CST) that interconnects the MST regions and single spanning trees.

  The spanning tree that is computed in a region appears as a subtree in the CST that encompasses the entire switched domain. The CIST is formed by the spanning-tree algorithm running among switches that support the IEEE 802.1w, IEEE 802.1s, and IEEE 802.1D standards. The CIST inside an MST region is the same as the CST outside a region.

Operations Within an Multiple Spanning Tree Region

The IST connects all the MSTP switches in a region. When the IST converges, the root of the IST becomes the CIST regional root. It is the switch within the region with the lowest device ID and path cost to the CIST root. The CIST regional root is also the CIST root if there is only one region in the network. If the CIST root is outside the region, one of the MSTP switches at the boundary of the region is selected as the CIST regional root.

When an MSTP switch initializes, it sends BPDUs claiming itself as the root of the CIST and the CIST regional root, with both of the path costs to the CIST root and to the CIST regional root set to zero. The switch also initializes all of its MST instances and claims to be the root for all of them. If the switch receives superior MST root information (lower device ID, lower path cost, and so forth) than currently stored for the port, it relinquishes its claim as the CIST regional root.

During initialization, a region might have many subregions, each with its own CIST regional root. As switches receive superior IST information, they leave their old subregions and join the new subregion that contains the true CIST regional root. All subregions shrink except for the one that contains the true CIST regional root.

For correct operation, all switches in the MST region must agree on the same CIST regional root. Therefore, any two switches in the region only synchronize their port roles for an MST instance if they converge to a common CIST regional root.

Operations Between Multiple Spanning Tree Regions

If there are multiple regions or legacy IEEE 802.1D switches within the network, MSTP establishes and maintains the CST, which includes all MST regions and all legacy STP switches in the network. The MST instances combine with the IST at the boundary of the region to become the CST.

The IST connects all the MSTP switches in the region and appears as a subtree in the CIST that encompasses the entire switched domain. The root of the subtree is the CIST regional root. The MST region appears as a virtual switch to adjacent STP switches and MST regions.

Only the CST instance sends and receives BPDUs, and MST instances add their spanning-tree information into the BPDUs to interact with neighboring switches and compute the final spanning-tree topology. Because of this, the spanning-tree parameters that are related to BPDU transmission (for example, hello time, forward time, max-age, and max-hops) are configured only on the CST instance but affect all MST instances. Parameters that are related to the spanning-tree topology (for example, switch priority, port VLAN cost, and port VLAN priority) can be configured on both the CST instance and the MST instance.

MSTP switches use Version 3 RSTP BPDUs or IEEE 802.1D STP BPDUs to communicate with legacy IEEE 802.1D devices. MSTP switches use MSTP BPDUs to communicate with MSTP devices.

IEEE 802.1s Terminology

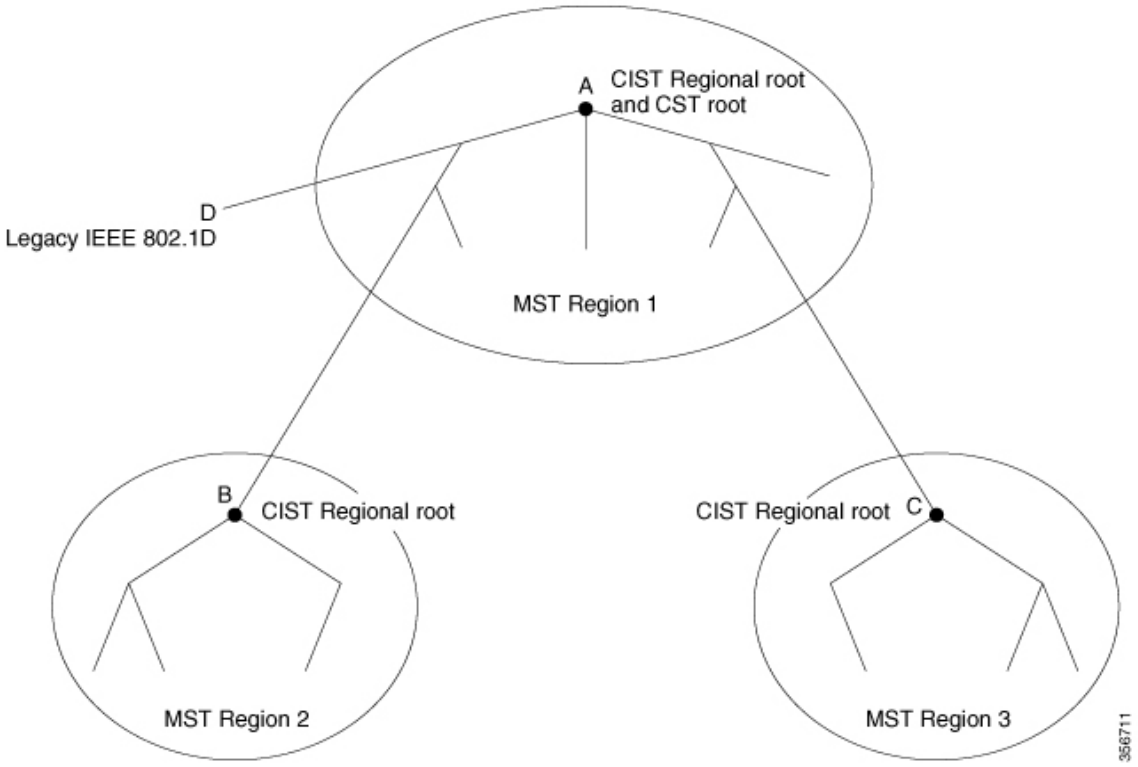
Some MST naming conventions that are used in Cisco's prestandard implementation have been changed to identify some *internal* or *regional* parameters. These parameters are significant only within an MST region, as opposed to external parameters that are relevant to the whole network. Because the CIST is the only spanning-tree instance that spans the whole network, only the CIST parameters require the external rather than the internal or regional qualifiers.

- The CIST root is the root switch for the unique instance that spans the whole network, the CIST.
- The CIST external root path cost is the cost to the CIST root. This cost is left unchanged within an MST region. Remember that an MST region looks like a single switch for the CIST. The CIST external root path cost is the root path cost that is calculated between these virtual devices and devices that do not belong to any region.
- If the CIST root is in the region, the CIST regional root is the CIST root. Otherwise, the CIST regional root is the closest switch to the CIST root in the region. The CIST regional root acts as a root switch for the IST.
- The CIST internal root path cost is the cost to the CIST regional root in a region. This cost is only relevant to the IST, instance 0.

Illustration of Multiple Spanning Tree Regions

This figure displays three MST regions and a legacy IEEE 802.1D device (D). The CIST regional root for region 1 (A) is also the CIST root. The CIST regional root for region 2 (B) and the CIST regional root for region 3 (C) are the roots for their respective subtrees within the CIST. The RSTP runs in all regions.

Figure 7: MST Regions, CIST Regional Root, and CST Root



Hop Count

The IST and MST instances do not use the message-age and maximum-age information in the configuration BPDU to compute the spanning-tree topology. Instead, they use the path cost to the root and a hop-count mechanism similar to the IP time-to-live (TTL) mechanism.

By using the **spanning-tree mst max-hops** global configuration command, you can configure the maximum hops inside the region and apply it to the IST and all MST instances in that region. The hop count achieves the same result as the message-age information (triggers a reconfiguration). The root switch of the instance always sends a BPDU (or M-record) with a cost of 0 and the hop count set to the maximum value. When a switch receives this BPDU, it decrements the received remaining hop count by one and propagates this value as the remaining hop count in the BPDUs it generates. When the count reaches zero, the switch discards the BPDU and ages the information that is held for the port.

The message-age and maximum-age information in the RSTP portion of the BPDU remain the same throughout the region, and the same values are propagated by the region designated ports at the boundary.

Boundary Ports

In the Cisco prestandard implementation, a boundary port connects an MST region to a single spanning-tree region running RSTP, to a single spanning-tree region running PVST+ or rapid PVST+, or to another MST region with a different MST configuration. A boundary port also connects to a LAN, the designated device of which is either a single spanning-tree switch or a switch with a different MST configuration.

There is no definition of a boundary port in the IEEE 802.1s standard. The IEEE 802.1Q-2002 standard identifies two kinds of messages that a port can receive:

- internal (coming from the same region)
- external (coming from another region)

When a message is internal, the CIST part is received by the CIST, and each MST instance receives its respective M-record.

When a message is external, it is received only by the CIST. If the CIST role is root or alternate, or if the external BPDU is a topology change, it could have an impact on the MST instances.

An MST region includes both devices and LANs. A segment belongs to the region of its designated port. Therefore, a port in a different region than the designated port for a segment is a boundary port. This definition allows two ports internal to a region to share a segment with a port belonging to a different region, creating the possibility of a port receiving both internal and external messages.

The primary change from the Cisco prestandard implementation is that a designated port is not defined as boundary, unless it is running in an STP-compatible mode.

Note If there is a legacy STP device on the segment, messages are always considered external.

The other change from the Cisco prestandard implementation is that the CIST regional root device ID field is now inserted where an RSTP or legacy IEEE 802.1Q device has the sender device ID. The whole region performs like a single virtual device by sending a consistent sender device ID to neighboring devices. In this example, Switch C would receive a BPDU with the same consistent sender device ID of root, whether or not A or B is designated for the segment.

IEEE 802.1s Implementation

The Cisco implementation of the IEEE MST standard includes features required to meet the standard, as well as some of the desirable prestandard functionality that is not yet incorporated into the published standard.

Port Role Naming Change

The boundary role is no longer in the final MST standard, but this boundary concept is maintained in Cisco's implementation. However, an MST instance port at a boundary of the region might not follow the state of the corresponding CIST port. Two boundary roles currently exist:

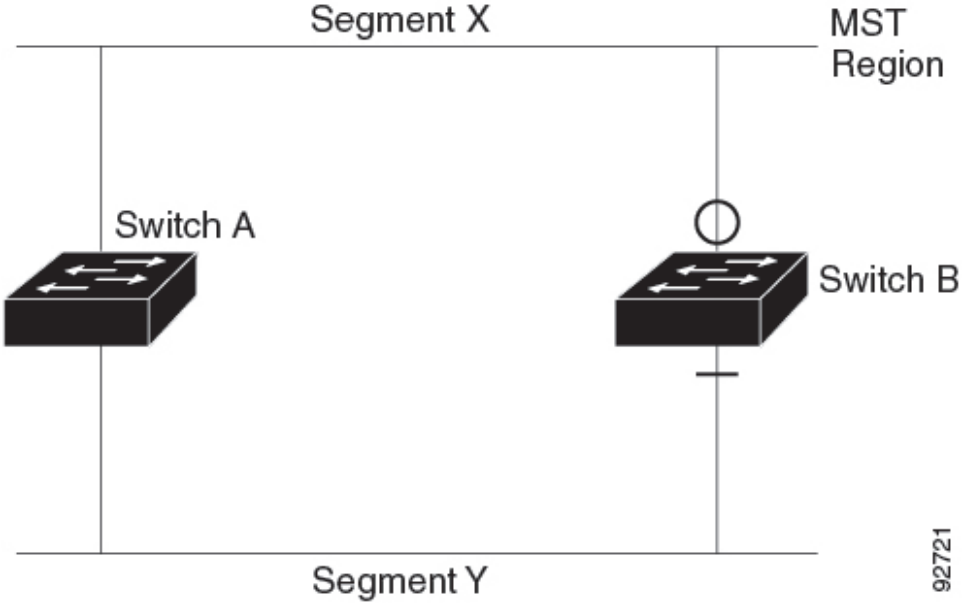
- The boundary port is the root port of the CIST regional root—When the CIST instance port is proposed and is in sync, it can send back an agreement and move to the forwarding state only after all the corresponding MSTI ports are in sync (and thus forwarding). The MSTI ports now have a special *primary* role.
- The boundary port is not the root port of the CIST regional root—The MSTI ports follow the state and role of the CIST port. The standard provides less information, and it might be difficult to understand why an MSTI port can be alternately blocking when it receives no BPDUs (MRecords). In this case, although the boundary role no longer exists, the **show** commands identify a port as boundary in the *type* column of the output.

Interoperation Between Legacy and Standard Devices

Because automatic detection of prestandard devices can fail, you can use an interface configuration command to identify prestandard ports. A region cannot be formed between a standard and a prestandard device, but they can interoperate by using the CIST. Only the capability of load-balancing over different instances is lost in that particular case. The CLI displays different flags depending on the port configuration when a port receives prestandard BPDUs. A syslog message also appears the first time a device receives a prestandard BPDU on a port that has not been configured for prestandard BPDU transmission.

Figure 8: Standard and Prestandard Device Interoperation

Assume that A is a standard switch and B a prestandard switch, both configured to be in the same region. A is the root switch for the CIST, and B has a root port (BX) on segment X and an alternate port (BY) on segment Y. If segment Y flaps, and the port on BY becomes the alternate before sending out a single prestandard BPDU, AY cannot detect that a prestandard switch is connected to Y and continues to send standard BPDUs. The port BY is fixed in a boundary, and no load balancing is possible between A and B. The same problem exists on segment X, but B might transmit topology changes.



Note We recommend that you minimize the interaction between standard and prestandard MST implementations.

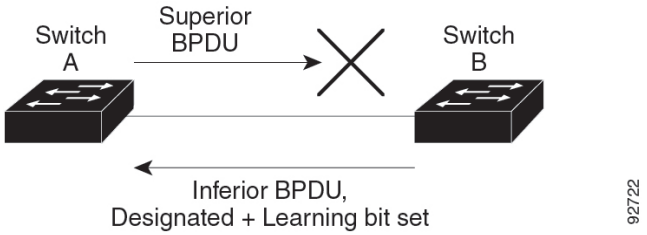
Detecting Unidirectional Link Failure

This feature is not yet present in the IEEE MST standard, but it is included in this Cisco IOS release. The software checks the consistency of the port role and state in the received BPDUs to detect unidirectional link failures that could cause bridging loops.

When a designated port detects a conflict, it keeps its role, but reverts to the discarding state because disrupting connectivity in case of inconsistency is preferable to opening a bridging loop.

Figure 9: Detecting Unidirectional Link Failure

This figure illustrates a unidirectional link failure that typically creates a bridging loop. Switch A is the root device, and its BPDUs are lost on the link leading to Switch B. RSTP and MST BPDUs include the role and state of the sending port. With this information, Switch A can detect that Switch B does not react to the superior BPDUs it sends and that Switch B is the designated, not root switch. As a result, Switch A blocks (or keeps blocking) its port, which prevents the bridging loop.



Interoperability with IEEE 802.1D Spanning Tree Protocol

A device running MSTP supports a built-in protocol migration mechanism that enables it to interoperate with legacy IEEE 802.1D devices. If this device receives a legacy IEEE 802.1D configuration BPDU (a BPDU with the protocol version set to 0), it sends only IEEE 802.1D BPDUs on that port. An MSTP device also can detect that a port is at the boundary of a region when it receives a legacy BPDU, an MSTP BPDU (Version 3) associated with a different region, or an RSTP BPDU (Version 2).

However, the device does not automatically revert to the MSTP mode if it no longer receives IEEE 802.1D BPDUs because it cannot detect whether the legacy switch has been removed from the link unless the legacy switch is the designated device. A device might also continue to assign a boundary role to a port when the device to which this device is connected has joined the region. To restart the protocol migration process (force the renegotiation with neighboring devices), use the **clear spanning-tree detected-protocols** privileged EXEC command.

If all the legacy switches on the link are RSTP devices, they can process MSTP BPDUs as if they are RSTP BPDUs. Therefore, MSTP devices send either a Version 0 configuration and TCN BPDUs or Version 3 MSTP BPDUs on a boundary port. A boundary port connects to a LAN, the designated device of which is either a single spanning-tree switch or a switch with a different MST configuration.

Rapid Spanning Tree Protocol Overview

The RSTP takes advantage of point-to-point wiring and provides rapid convergence of the spanning tree. Reconfiguration of the spanning tree can occur in less than 1 second (in contrast to 50 seconds with the default settings in the IEEE 802.1D spanning tree).

Port Roles and the Active Topology

The RSTP provides rapid convergence of the spanning tree by assigning port roles and by learning the active topology. The RSTP builds upon the IEEE 802.1D STP to select the device with the highest device priority (lowest numerical priority value) as the root device. The RSTP then assigns one of these port roles to individual ports:

- Root port—Provides the best path (lowest cost) when the device forwards packets to the root switch.

- Designated port—Connects to the designated device, which incurs the lowest path cost when forwarding packets from that LAN to the root switch. The port through which the designated device is attached to the LAN is called the designated port.
- Alternate port—Offers an alternate path toward the root switch to that provided by the current root port.
- Backup port—Acts as a backup for the path that is provided by a designated port toward the leaves of the spanning tree. A backup port can exist only when two ports are connected in a loopback by a point-to-point link or when a device has two or more connections to a shared LAN segment.
- Disabled port—Has no role within the operation of the spanning tree.

A port with the root or a designated port role is included in the active topology. A port with the alternate or backup port role is excluded from the active topology.

In a stable topology with consistent port roles throughout the network, the RSTP ensures that every root port and designated port immediately transition to the forwarding state while all alternate and backup ports are always in the discarding state (equivalent to blocking in IEEE 802.1D). The port state controls the operation of the forwarding and learning processes.

Table 18: Port State Comparison

Operational Status	STP Port State (IEEE 802.1D)	RSTP Port State	Is Port Included in the Active Topology?
Enabled	Blocking	Discarding	No
Enabled	Listening	Discarding	No
Enabled	Learning	Learning	Yes
Enabled	Forwarding	Forwarding	Yes
Disabled	Disabled	Discarding	No

To be consistent with Cisco STP implementations, this guide defines the port state as *blocking* instead of *discarding*. Designated ports start in the listening state.

Rapid Convergence

The RSTP provides for rapid recovery of connectivity following the failure of a device, a device port, or a LAN. It provides rapid convergence for edge ports, new root ports, and ports connected through point-to-point links as follows:

- Edge ports—If you configure a port as an edge port on an RSTP device by using the **spanning-tree portfast** interface configuration command, the edge port immediately transitions to the forwarding state. An edge port is the same as a Port Fast-enabled port, and you should enable it only on ports that connect to a single end station.
- Root ports—If the RSTP selects a new root port, it blocks the old root port and immediately transitions the new root port to the forwarding state.
- Point-to-point links—If you connect a port to another port through a point-to-point link and the local port becomes a designated port, it negotiates a rapid transition with the other port by using the proposal-agreement handshake to ensure a loop-free topology.

Figure 10: Proposal and Agreement Handshaking for Rapid Convergence

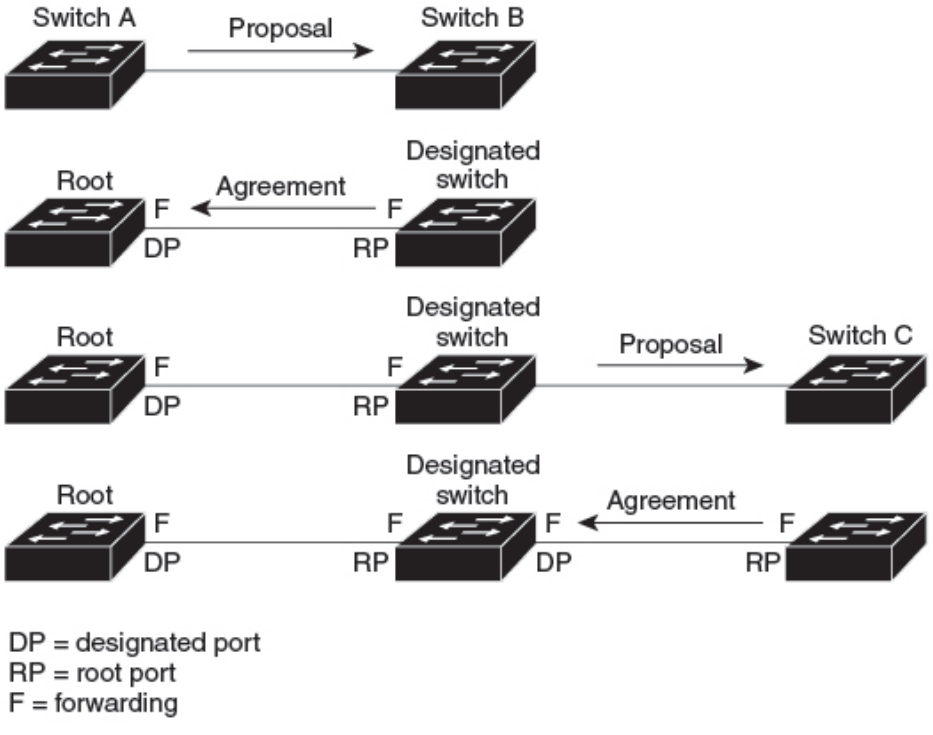
Switch A is connected to Switch B through a point-to-point link, and all of the ports are in the blocking state. Assume that the priority of Switch A is a smaller numerical value than the priority of Switch B. Switch A sends a proposal message (a configuration BPDU with the proposal flag set) to Switch B, proposing itself as the designated device.

After receiving the proposal message, Switch B selects as its new root port the port from which the proposal message was received, forces all nonedge ports to the blocking state, and sends an agreement message (a BPDU with the agreement flag set) through its new root port.

After receiving Switch B's agreement message, Switch A also immediately transitions its designated port to the forwarding state. No loops in the network are formed because Switch B blocked all of its nonedge ports and because there is a point-to-point link between Switch A and B.

When Switch C is connected to Switch B, a similar set of handshaking messages are exchanged. Switch C selects the port connected to Switch B as its root port, and both ends immediately transition to the forwarding state. With each iteration of this handshaking process, one more device joins the active topology. As the network converges, this proposal-agreement handshaking progresses from the root toward the leaves of the spanning tree.

The device learns the link type from the port duplex mode: a full-duplex port is considered to have a point-to-point connection; a half-duplex port is considered to have a shared connection. You can override the default setting that is controlled by the duplex setting by using the **spanning-tree link-type** interface configuration command.



Synchronization of Port Roles

When the device receives a proposal message on one of its ports and that port is selected as the new root port, the RSTP forces all other ports to synchronize with the new root information.

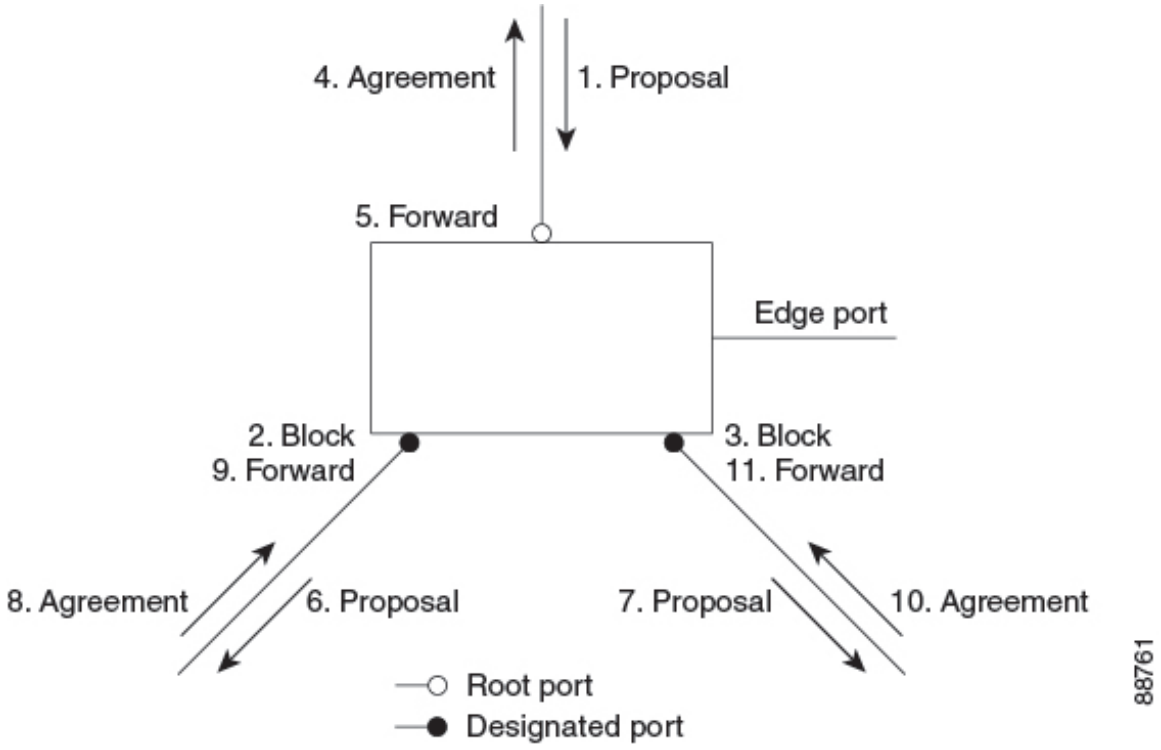
The device is synchronized with superior root information that is received on the root port if all other ports are synchronized. An individual port on the device is synchronized if:

- That port is in the blocking state.
- It is an edge port (a port that is configured to be at the edge of the network).

If a designated port is in the forwarding state and is not configured as an edge port, it transitions to the blocking state when the RSTP forces it to synchronize with new root information. In general, when the RSTP forces a port to synchronize with root information and the port does not satisfy any of the above conditions, its port state is set to blocking.

Figure 11: Sequence of Events During Rapid Convergence

After ensuring that all of the ports are synchronized, the device sends an agreement message to the designated device corresponding to its root port. When the devices that are connected by a point-to-point link are in agreement about their port roles, the RSTP immediately transitions the port states to forwarding.



Bridge Protocol Data Unit Format and Processing

The RSTP BPDU format is the same as the IEEE 802.1D BPDU format except that the protocol version is set to 2. A new 1-byte Version 1 Length field is set to zero, which means that no version 1 protocol information is present.

Table 19: RSTP BPDU Flags

Bit	Function
0	Topology change (TC)
1	Proposal
2–3: 00 01 10 11	Port role: Unknown Alternate port Root port Designated port
4	Learning
5	Forwarding
6	Agreement
7	Topology change acknowledgement (TCA)

The sending device sets the proposal flag in the RSTP BPDU to propose itself as the designated device on that LAN. The port role in the proposal message is always set to the designated port.

The sending device sets the agreement flag in the RSTP BPDU to accept the previous proposal. The port role in the agreement message is always set to the root port.

The RSTP does not have a separate topology change notification (TCN) BPDU. It uses the topology change (TC) flag to show the topology changes. However, for interoperability with IEEE 802.1D devices, the RSTP device processes and generates TCN BPDUs.

The learning and forwarding flags are set according to the state of the sending port.

Processing Superior Bridge Protocol Data Unit Information

If a port receives superior root information (lower device ID, lower path cost, and so forth) than currently stored for the port, the RSTP triggers a reconfiguration. If the port is proposed and is selected as the new root port, RSTP forces all the other ports to synchronize.

If the BPDU received is an RSTP BPDU with the proposal flag set, the device sends an agreement message after all of the other ports are synchronized. If the BPDU is an IEEE 802.1D BPDU, the device does not set the proposal flag and starts the forward-delay timer for the port. The new root port requires twice the forward-delay time to transition to the forwarding state.

If the superior information that is received on the port causes the port to become a backup or alternate port, RSTP sets the port to the blocking state but does not send the agreement message. The designated port continues sending BPDUs with the proposal flag set until the forward-delay timer expires, at which time the port transitions to the forwarding state.

Processing Inferior Bridge Protocol Data Unit Information

If a designated port receives an inferior BPDU (such as a higher device ID or a higher path cost than currently stored for the port) with a designated port role, it immediately replies with its own information.

Topology Changes

This section describes the differences between the RSTP and the IEEE 802.1D in handling spanning-tree topology changes.

- Detection—Unlike IEEE 802.1D in which *any* transition between the blocking and the forwarding state causes a topology change, *only* transitions from the blocking to the forwarding state cause a topology change with RSTP (only an increase in connectivity is considered a topology change). State changes on an edge port do not cause a topology change. When an RSTP device detects a topology change, it deletes the learned information on all of its nonedge ports except on those from which it received the TC notification.
- Notification—Unlike IEEE 802.1D, which uses TCN BPDUs, the RSTP does not use them. However, for IEEE 802.1D interoperability, an RSTP device processes and generates TCN BPDUs.
- Acknowledgement—When an RSTP device receives a TCN message on a designated port from an IEEE 802.1D device, it replies with an IEEE 802.1D configuration BPDU with the TCA bit set. However, if the TC-while timer (the same as the topology-change timer in IEEE 802.1D) is active on a root port that is connected to an IEEE 802.1D device and a configuration BPDU with the TCA bit set is received, the TC-while timer is reset.

  This behavior is only required to support IEEE 802.1D devices. The RSTP BPDUs never have the TCA bit set.
- Propagation—When an RSTP device receives a TC message from another device through a designated or root port, it propagates the change to all of its nonedge, designated ports and to the root port (excluding the port on which it is received). The device starts the TC-while timer for all such ports and flushes the information learned on them.
- Protocol migration—For backward compatibility with IEEE 802.1D devices, RSTP selectively sends IEEE 802.1D configuration BPDUs and TCN BPDUs on a per-port basis.

  When a port is initialized, the migrate-delay timer is started (specifies the minimum time during which RSTP BPDUs are sent), and RSTP BPDUs are sent. While this timer is active, the device processes all BPDUs received on that port and ignores the protocol type.

  If the device receives an IEEE 802.1D BPDU after the port migration-delay timer has expired, it assumes that it is connected to an IEEE 802.1D device and starts using only IEEE 802.1D BPDUs. However, if the RSTP device is using IEEE 802.1D BPDUs on a port and receives an RSTP BPDU after the timer has expired, it restarts the timer and starts using RSTP BPDUs on that port.

Protocol Migration Process

A device running MSTP supports a built-in protocol migration mechanism that enables it to interoperate with legacy IEEE 802.1D devices. If this device receives a legacy IEEE 802.1D configuration BPDU (a BPDU with the protocol version set to 0), it sends only IEEE 802.1D BPDUs on that port. An MSTP device also can detect that a port is at the boundary of a region when it receives a legacy BPDU, an MST BPDU (Version 3) associated with a different region, or an RST BPDU (Version 2).

However, the device does not automatically revert to the MSTP mode if it no longer receives IEEE 802.1D BPDUs because it cannot detect whether the legacy switch has been removed from the link unless the legacy switch is the designated device. A device also might continue to assign a boundary role to a port when the device to which it is connected has joined the region.

Default Multiple Spanning Tree Protocol Configuration

Table 20: Default MSTP Configuration

Feature	Default Setting
Spanning-tree mode	MSTP
Device priority (configurable on a per-CIST port basis)	32768
Spanning-tree port priority (configurable on a per-CIST port basis)	128
Spanning-tree port cost (configurable on a per-CIST port basis)	• 1000Mb/s: 4 • 100Mb/s: 19 • 10Mb/s: 100
Hello time	2 seconds
Forward-delay time	15
Maximum-aging time	20 seconds
Maximum hop count	20 hops

How to Configure Multiple Spanning Tree Protocol and Parameters

The following sections provide information about configuring MSTP and MSTP parameters:

Specifying the Multiple Spanning Tree Region Configuration and Enabling Multiple Spanning Tree Protocol

For two or more switches to be in the same MST region, they must have the same VLAN-to-instance mapping, the same configuration revision number, and the same name.

A region can have one member or multiple members with the same MST configuration; each member must be capable of processing RSTP BPDUs. There is no limit to the number of MST regions in a network, but each region can only support up to 64 spanning-tree instances. You can assign a VLAN to only one spanning-tree instance at a time.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst configuration** **Example:** `Device(config)# spanning-tree mst configuration`	Enters MST configuration mode.
Step 4	**instance** *instance-id* **vlan** *vlan-range* **Example:** `Device(config-mst)# instance 1 vlan 10-20`	Maps VLANs to an MST instance. • For *instance-id*, the range is 0 to 4094. • For **vlan** *vlan-range*, the range is 1 to 4094. When you map VLANs to an MST instance, the mapping is incremental, and the VLANs specified in the command are added to or removed from the VLANs that were previously mapped. To specify a VLAN range, use a hyphen; for example, **instance 1 vlan 1-63** maps VLANs 1 through 63 to MST instance 1. To specify a VLAN series, use a comma; for example, **instance 1 vlan 10, 20, 30** maps VLANs 10, 20, and 30 to MST instance 1.
Step 5	**name** *name* **Example:** `Device(config-mst)# name region1`	Specifies the configuration name. The *name* string has a maximum length of 32 characters and is case sensitive.
Step 6	**revision** *version* **Example:** `Device(config-mst)# revision 1`	Specifies the configuration revision number. The range is 0 to 65535.
Step 7	**show pending** **Example:** `Device(config-mst)# show pending`	Verifies your configuration by displaying the pending configuration.
Step 8	**exit** **Example:** `Device(config-mst)# exit`	Applies all changes, and returns to global configuration mode.

	Command or Action	Purpose
Step 9	**spanning-tree mode mst** **Example:** `Device(config)# spanning-tree mode mst`	Enables MSTP. RSTP is also enabled. Changing spanning-tree modes can disrupt traffic because all spanning-tree instances are stopped for the previous mode and restarted in the new mode. You cannot run both MSTP and PVST+ or both MSTP and Rapid PVST+ at the same time.
Step 10	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Root Device

To configure the root device, perform this procedure:

Before you begin

- An MST must be specified and enabled on the device. .
- You must also know the specified MST instance ID.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst** *instance-id* **root primary** **Example:** `Device(config)# spanning-tree mst 0 root primary`	Configures a device as the root device. For *instance-id*, you can specify a single instance, a range of instances separated by a hyphen, or a series of instances separated by a comma. The range is 0 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring a Secondary Root Device

When you configure a device with the extended system ID support as the secondary root, the device priority is modified from the default value (32768) to 28672. The device is then likely to become the root device for the specified instance if the primary root device fails. This is assuming that the other network devices use the default device priority of 32768 and therefore are unlikely to become the root device.

You can execute this command on more than one device to configure multiple backup root devices. Use the same network diameter and hello-time values that you used when you configured the primary root device with the **spanning-tree mst** *instance-id* **root primary** global configuration command.

To configure a secondary root device, perform this procedure:

Before you begin

- An MST must be specified and enabled on the device.
- You must also know the specified MST instance ID.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst** *instance-id* **root secondary** **Example:** `Device(config)# spanning-tree mst 0 root secondary`	Configures a devices as the secondary root device. For *instance-id*, you can specify a single instance, a range of instances separated by a hyphen, or a series of instances separated by a comma. The range is 0 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring Port Priority

If a loop occurs, the MSTP uses the port priority when selecting an interface to put into the forwarding state. You can assign higher priority values (lower numerical values) to interfaces that you want selected first and lower priority values (higher numerical values) that you want selected last. If all interfaces have the same priority value, the MSTP puts the interface with the lowest interface number in the forwarding state and blocks the other interfaces.

To configure port priority, perform this procedure:

Before you begin

- An MST must be specified and enabled on the device.
- You must also know the specified MST instance ID and the interface used.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface to configure, and enters interface configuration mode.
Step 4	**spanning-tree mst** *instance-id* **port-priority** *priority* **Example:** `Device(config-if)# spanning-tree mst 0 port-priority 64`	Configures port priority. • For *instance-id*, you can specify a single instance, a range of instances separated by a hyphen, or a series of instances separated by a comma. The range is 0 to 4094. • For *priority*, the range is 0 to 240 in increments of 16. The default is 128. The lower the number, the higher the priority. The priority values are 0, 16, 32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, 224, and 240. All other values are rejected.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

The **show spanning-tree mst interface** *interface-id* privileged EXEC command displays information only if the port is in a link-up operative state. Otherwise, you can use the **show running-config interface** privileged EXEC command to confirm the configuration.

(Optional) Configuring Path Cost

The MSTP path cost default value is derived from the media speed of an interface. If a loop occurs, the MSTP uses cost when selecting an interface to put in the forwarding state. You can assign lower cost values to interfaces that you want selected first and higher cost values that you want selected last. If all interfaces have the same cost value, the MSTP puts the interface with the lowest interface number in the forwarding state and blocks the other interfaces.

To configure path cost, perform this procedure:

Before you begin

- An MST must be specified and enabled on the device.
- You must also know the specified MST instance ID and the interface used.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports and port-channel logical interfaces. The port-channel range is 1 to 48.
Step 4	**spanning-tree mst** *instance-id* **cost** *cost* **Example:** `Device(config-if)# spanning-tree mst 0 cost 17031970`	Configures the cost. If a loop occurs, the MSTP uses the path cost when selecting an interface to place into the forwarding state. A lower path cost represents higher-speed transmission. • For *instance-id*, you can specify a single instance, a range of instances separated by a hyphen, or a series of instances separated by a comma. The range is 0 to 4094. • For *cost*, the range is 1 to 200000000; the default value is derived from the media speed of the interface.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	

The **show spanning-tree mst interface** *interface-id* privileged EXEC command displays information only for ports that are in a link-up operative state. Otherwise, you can use the **show running-config** privileged EXEC command to confirm the configuration.

(Optional) Configuring the Device Priority

Changing the priority of a device makes it more likely to be chosen as the root switch whether it is a standalone switch.

Note Exercise care when using this command. For normal network configurations, we recommend that you use the **spanning-tree mst** i*nstance-id* **root primary** and the **spanning-tree mst** *instance-id* **root secondary** global configuration commands to specify a device as the root or secondary root device. You should modify the device priority only in circumstances where these commands do not work.

To configure the device priority, perform this procedure:

Before you begin

- An MST must be specified and enabled on the device.
- You must also know the specified MST instance ID used.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst** *instance-id* **priority** *priority* **Example:** `Device(config)# spanning-tree mst 0 priority 40960`	Configures the device priority. • For *instance-id*, you can specify a single instance, a range of instances separated by a hyphen, or a series of instances separated by a comma. The range is 0 to 4094. • For *priority*, the range is 0 to 61440 in increments of 4096; the default is 32768. The lower the number, the more likely the device will be chosen as the root switch.

	Command or Action	Purpose
		Priority values are 0, 4096, 8192, 12288, 16384, 20480, 24576, 28672, 32768, 36864, 40960, 45056, 49152, 53248, 57344, and 61440. These are the only acceptable values.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Hello Time

The hello time is the time interval between configuration messages that are generated and sent by the root device.

To configure the hello time, perform this procedure:

Before you begin

An MST must be specified and enabled on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst hello-time** *seconds* **Example:** `Device(config)# spanning-tree mst hello-time 4`	Configures the hello time for all MST instances. The hello time is the time interval between configuration messages that are generated and sent by the root device. These messages indicate that the device is alive. For *seconds*, the range is 1 to 10; the default is 3.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Forwarding-Delay Time

To configure the forwarding-delay time, perform this procedure:

Before you begin

An MST must be specified and enabled on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst forward-time** *seconds* **Example:** `Device(config)# spanning-tree mst forward-time 25`	Configures the forward time for all MST instances. The forwarding delay is the number of seconds a port waits before changing from its spanning-tree learning and listening states to the forwarding state. For *seconds*, the range is 4 to 30; the default is 20.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Maximum-Aging Time

To configure the maximum-aging time, perform this procedure:

Before you begin

An MST must be specified and enabled on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst max-age** *seconds* **Example:** `Device(config)# spanning-tree mst max-age 40`	Configures the maximum-aging time for all MST instances. The maximum-aging time is the number of seconds a device waits without receiving spanning-tree configuration messages before attempting a reconfiguration. For *seconds*, the range is 6 to 40; the default is 20.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Maximum-Hop Count

To configure the maximum-hop count, perform this procedure:

Before you begin

An MST must be specified and enabled on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree mst max-hops** *hop-count* **Example:** `Device(config)# spanning-tree mst max-hops 25`	Specifies the number of hops in a region before the BPDU is discarded, and the information that is held for a port is aged. For *hop-count*, the range is 1 to 255; the default is 20.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Specifying the Link Type to Ensure Rapid Transitions

If you connect a port to another port through a point-to-point link and the local port becomes a designated port, the RSTP negotiates a rapid transition with the other port by using the proposal-agreement handshake to ensure a loop-free topology.

By default, the link type is controlled from the duplex mode of the interface: a full-duplex port is considered to have a point-to-point connection; a half-duplex port is considered to have a shared connection. If you have a half-duplex link physically connected point-to-point to a single port on a remote device running MSTP, you can override the default setting of the link type and enable rapid transitions to the forwarding state.

To specify the link type to ensure rapid transitions, perform this procedure:

Before you begin

- An MST must be specified and enabled on the device.
- You must also know the specified MST instance ID and the interface used.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports, VLANs, and port-channel logical interfaces. The VLAN ID range is 1 to 4094. The port-channel range is 1 to 48.
Step 4	**spanning-tree link-type point-to-point** **Example:** `Device(config-if)# spanning-tree link-type point-to-point`	Specifies that the link type of a port is point-to-point.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

(Optional) Designating the Neighbor Type

A topology could contain both prestandard and IEEE 802.1s standard compliant devices. By default, ports can automatically detect prestandard devices, but they can still receive both standard and prestandard BPDUs. When there is a mismatch between a device and its neighbor, only the CIST runs on the interface.

You can choose to set a port to send only prestandard BPDUs. The prestandard flag appears in all the **show** commands, even if the port is in STP compatibility mode.

To designate the neighbor type, perform this procedure:

Before you begin

An MST must be specified and enabled on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface to configure, and enters interface configuration mode. Valid interfaces include physical ports.
Step 4	**spanning-tree mst pre-standard** **Example:** `Device(config-if)# spanning-tree mst pre-standard`	Specifies that the port can send only prestandard BPDUs.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Restarting the Protocol Migration Process

This procedure restarts the protocol migration process and forces renegotiation with neighboring devices. It reverts the device to MST mode. It is needed when the device no longer receives IEEE 802.1D BPDUs after it has been receiving them.

Follow these steps to restart the protocol migration process (force the renegotiation with neighboring devices) on the device.

Before you begin

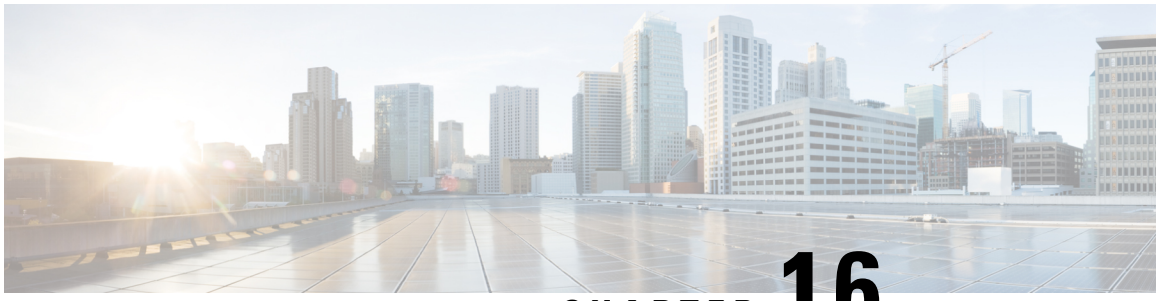
- An MST must be specified and enabled on the device.
- If you want to use the interface version of the command, you must also know the MST interface used.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	Enter one of the following commands: • **clear spanning-tree detected-protocols** • **clear spanning-tree detected-protocols interface** *interface-id* **Example:** `Device# clear spanning-tree detected-protocols` or `Device# clear spanning-tree detected-protocols interface gigabitethernet 1/1`	The device reverts to the MSTP mode, and the protocol migration process restarts.

What to do next

This procedure may need to be repeated if the device receives more legacy IEEE 802.1D configuration BPDUs (BPDUs with the protocol version set to 0).



CHAPTER 16

Configuring Optional Spanning-Tree Features

Information About Optional Spanning-Tree Features

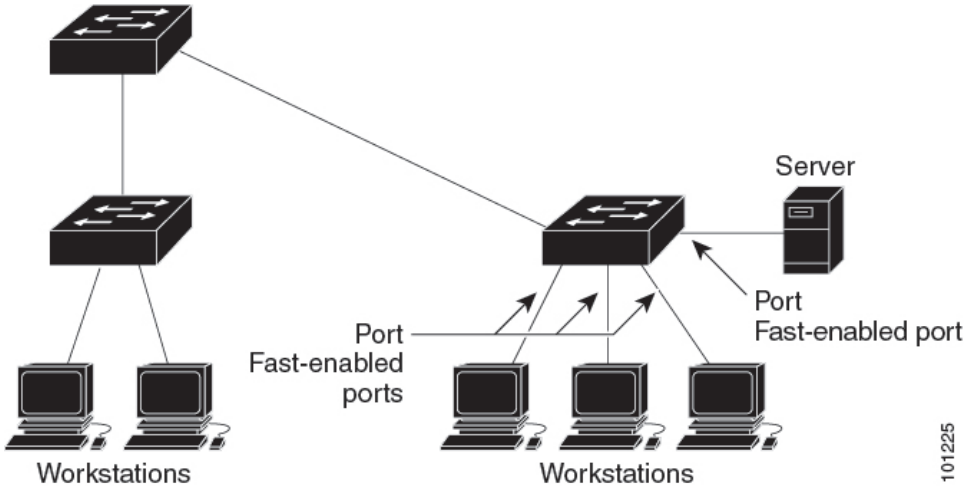
The following sections provide information about Optional Spanning-Tree features:

PortFast

PortFast immediately brings an interface that is configured as an access or trunk port to the forwarding state from a blocking state, bypassing the listening and learning states.

Figure 12: PortFast-Enabled Interfaces

You can use PortFast on interfaces that are connected to a single workstation or server to allow those devices to immediately connect to the network, rather than waiting for the spanning tree to converge.



Interfaces connected to a single workstation or server should not receive bridge protocol data units (BPDUs). An interface with PortFast enabled goes through the normal cycle of spanning-tree status changes when the switch is restarted.

You can enable this feature by enabling it on either the interface or on all nontrunking ports.

Bridge Protocol Data Unit Guard

The Bridge Protocol Data Unit (BPDU) guard feature can be globally enabled on the switch or can be enabled per port, but the feature operates with some differences.

When you enable BPDU guard at the global level on PortFast enabled ports, spanning tree shuts down ports that are in a PortFast operational state if any BPDU is received on them. In a valid configuration, PortFast enabled ports do not receive BPDUs. Receiving a BPDU on a PortFast enabled port means an invalid configuration, such as the connection of an unauthorized device, and the BPDU guard feature puts the port in the error-disabled state. When this happens, the switch shuts down the entire port on which the violation occurred.

When you enable BPDU guard at the interface level on any port without also enabling the PortFast feature, and the port receives a BPDU, it is put in the error-disabled state.

The BPDU guard feature provides a secure response to invalid configurations because you must manually put the interface back in service. Use the BPDU guard feature in a service-provider network to prevent an access port from participating in the spanning tree.

Bridge Protocol Data Unit Filtering

The BPDU filtering feature can be globally enabled on the switch or can be enabled per interface, but the feature operates with some differences.

Enabling BPDU filtering on PortFast enabled interfaces at the global level keeps those interfaces that are in a PortFast operational state from sending or receiving BPDUs. The interfaces still send a few BPDUs at link-up before the switch begins to filter outbound BPDUs. You should globally enable BPDU filtering on a switch so that hosts that are connected to these interfaces do not receive BPDUs. If a BPDU is received on a PortFast enabled interface, the interface loses its PortFast operational status, and BPDU filtering is disabled.

Enabling BPDU filtering on an interface without also enabling the PortFast feature keeps the interface from sending or receiving BPDUs.

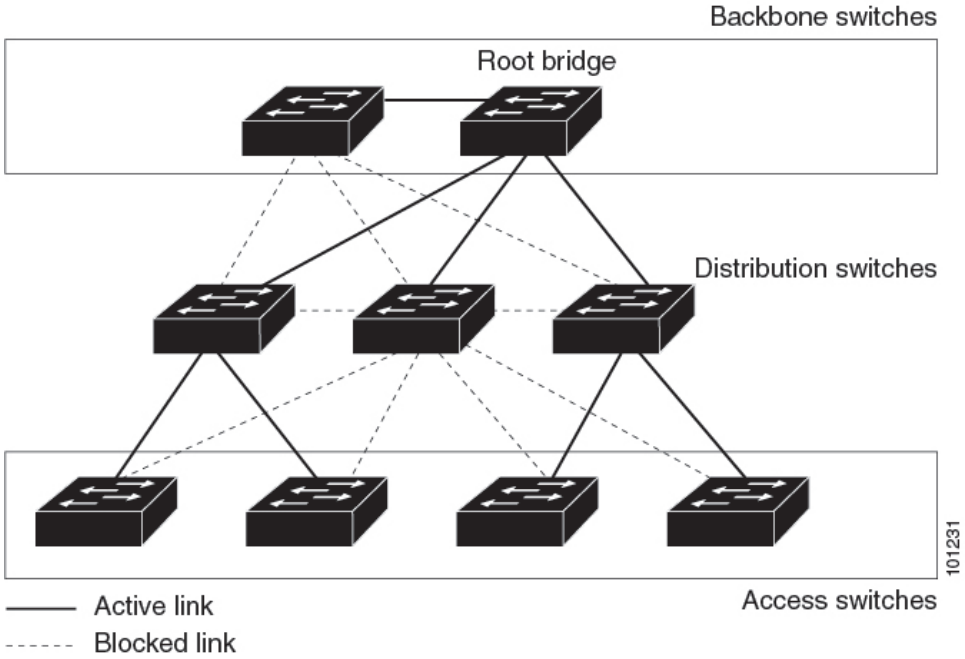
Caution Enabling BPDU filtering on an interface is the same as disabling spanning tree on it and can result in spanning-tree loops.

You can enable the BPDU filtering feature for the entire switch or for an interface.

UplinkFast

Figure 13: Switches in a Hierarchical Network

Switches in hierarchical networks can be grouped into backbone switches, distribution switches, and access switches. This complex network has distribution switches and access switches that each have at least one redundant link that spanning tree blocks to prevent loops.



If a switch loses connectivity, it begins using the alternate paths when the spanning tree selects a new root port. You can accelerate the choice of a new root port when a link or switch fails or when the spanning tree reconfigures itself by enabling UplinkFast. The root port transitions to the forwarding state immediately without going through the listening and learning states, as it would with the normal spanning-tree procedures.

When the spanning tree reconfigures the new root port, other interfaces flood the network with multicast packets, one for each address that was learned on the interface. You can limit these bursts of multicast traffic by reducing the max-update-rate parameter (the default for this parameter is 150 packets per second). However, if you enter zero, station-learning frames are not generated, so the spanning-tree topology converges more slowly after a loss of connectivity.

Note UplinkFast is most useful in wiring-closet switches at the access or edge of the network. It is not appropriate for backbone devices. This feature might not be useful for other types of applications.

UplinkFast provides fast convergence after a direct link failure and achieves load-balancing between redundant Layer 2 links using uplink groups. An uplink group is a set of Layer 2 interfaces (per VLAN), only one of which is forwarding at any given time. Specifically, an uplink group consists of the root port (which is forwarding) and a set of blocked ports, except for self-looping ports. The uplink group provides an alternate path in case the currently forwarding link fails.

Figure 14: UplinkFast Example Before Direct Link Failure

This topology has no link failures. Switch A, the root switch, is connected directly to Switch B over link L1 and to Switch C over link L2. The Layer 2 interface on Switch C that is connected directly to Switch B is in a blocking state.

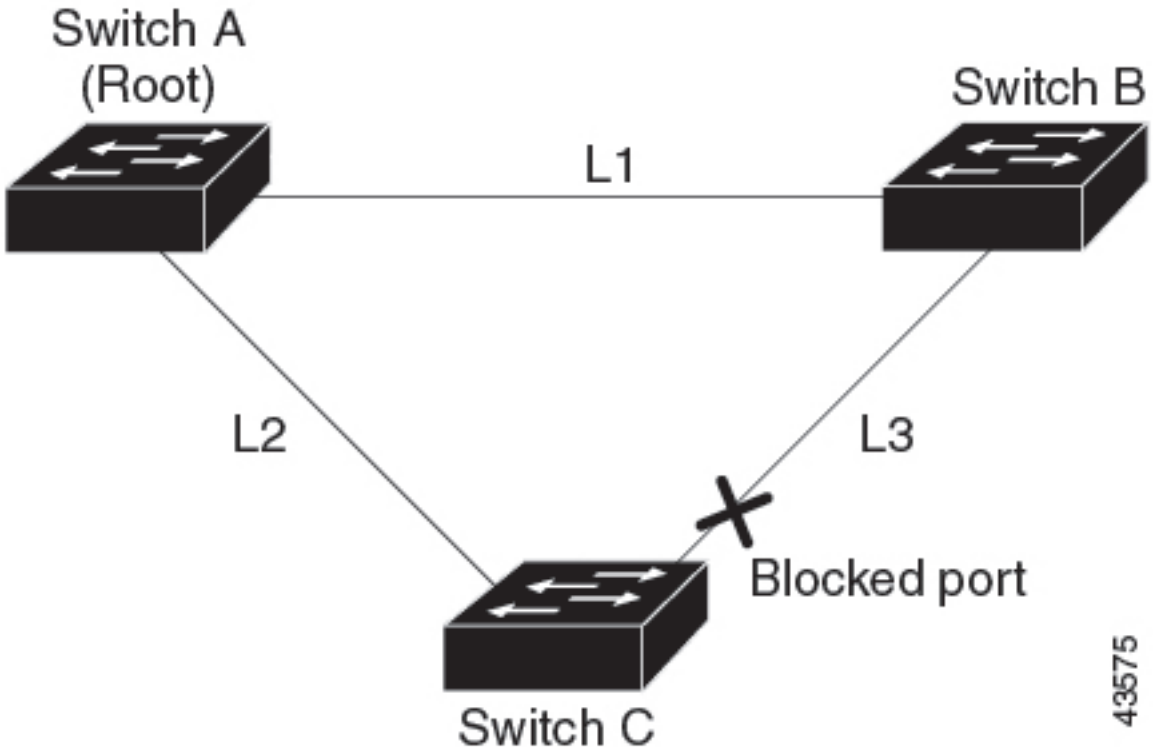
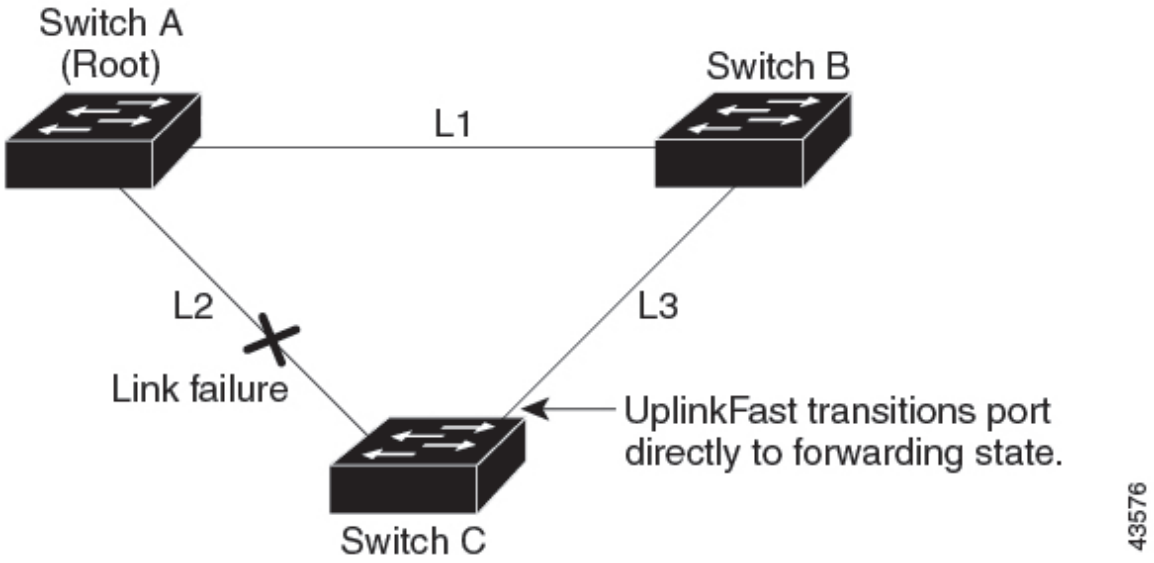


Figure 15: UplinkFast Example After Direct Link Failure

If Switch C detects a link failure on the currently active link L2 on the root port (a direct link failure), UplinkFast unblocks the blocked interface on Switch C and transitions it to the forwarding state without going through the listening and learning states. This change takes approximately 1 to 5 seconds.



BackboneFast

BackboneFast detects indirect failures in the core of the backbone. BackboneFast is a complementary technology to the UplinkFast feature, which responds to failures on links that are directly connected to access switches. BackboneFast optimizes the maximum-age timer, which controls the amount of time the switch stores protocol information that is received on an interface. When a switch receives an inferior BPDU from the designated port of another switch, the BPDU is a signal that the other switch might have lost its path to the root, and BackboneFast tries to find an alternate path to the root.

BackboneFast starts when a root port or blocked interface on a switch receives inferior BPDUs from its designated switch. An inferior BPDU identifies a switch that declares itself as both the root bridge and the designated switch. When a switch receives an inferior BPDU, it means that a link to which the switch is not directly connected (an indirect link) has failed (that is, the designated switch has lost its connection to the root switch). Under spanning-tree rules, the switch ignores inferior BPDUs for the maximum aging time (default is 20 seconds).

The switch tries to find if it has an alternate path to the root switch. If the inferior BPDU arrives on a blocked interface, the root port and other blocked interfaces on the switch become alternate paths to the root switch. (Self-looped ports are not considered alternate paths to the root switch.) If the inferior BPDU arrives on the root port, all blocked interfaces become alternate paths to the root switch. If the inferior BPDU arrives on the root port and there are no blocked interfaces, the switch assumes that it has lost connectivity to the root switch, causes the maximum aging time on the root port to expire, and becomes the root switch according to normal spanning-tree rules.

If the switch has alternate paths to the root switch, it uses these alternate paths to send a root link query (RLQ) request. The switch sends the RLQ request on all alternate paths to learn of an alternate root to the root switch and waits for an RLQ reply from other switches in the network. The switch sends the RLQ request on all alternate paths and waits for an RLQ reply from other switches in the network.

If the switch discovers that it still has an alternate path to the root, it expires the maximum aging time on the interface that received the inferior BPDU. If all the alternate paths to the root switch indicate that the switch has lost connectivity to the root switch, the switch expires the maximum aging time on the interface that received the RLQ reply. If one or more alternate paths can still connect to the root switch, the switch makes all interfaces on which it received an inferior BPDU its designated ports and moves them from the blocking state (if they were in the blocking state), through the listening and learning states, and into the forwarding state.

Figure 16: BackboneFast Example Before Indirect Link Failure

This is an example topology with no link failures. Switch A, the root switch, connects directly to Switch B over link L1 and to Switch C over link L2. The Layer 2 interface on Switch C that connects directly to Switch B is in the blocking state.

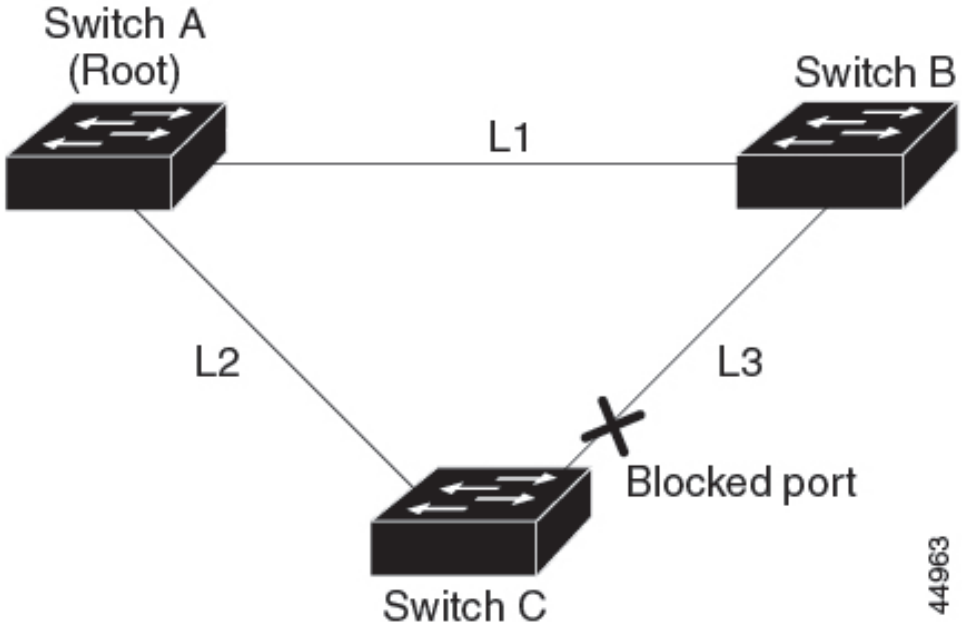


Figure 17: BackboneFast Example After Indirect Link Failure

If link L1 fails, Switch C cannot detect this failure because it is not connected directly to link L1. However, because Switch B is directly connected to the root switch over L1, it detects the failure, elects itself the root, and begins sending BPDUs to Switch C, identifying itself as the root. When Switch C receives the inferior BPDUs from Switch B, Switch C assumes that an indirect failure has occurred. At that point, BackboneFast allows the blocked interface on Switch C to move immediately to the listening state without waiting for the maximum aging time for the interface to expire. BackboneFast then transitions the Layer 2 interface on Switch C to the forwarding state, providing a path from Switch B to Switch A. The root-switch election takes approximately 30 seconds, twice the Forward Delay time if the default Forward Delay time of 15 seconds is set. BackboneFast reconfigures the topology to account for the failure of link L1.

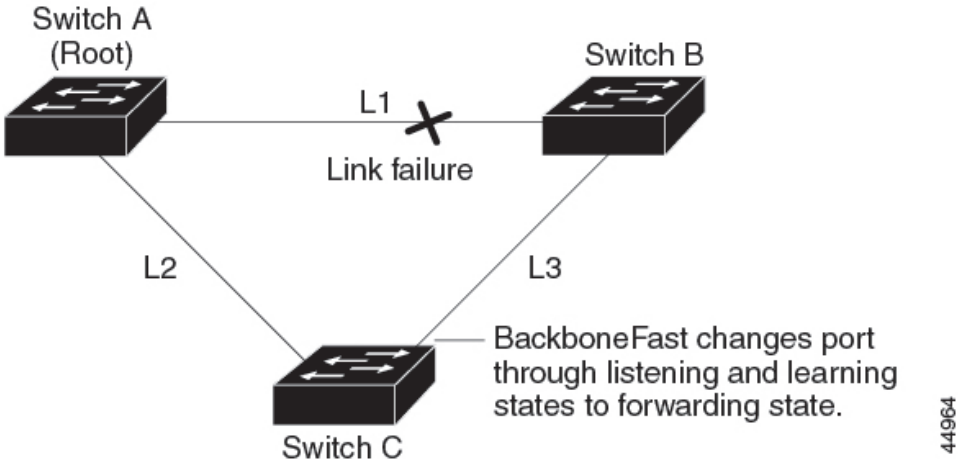
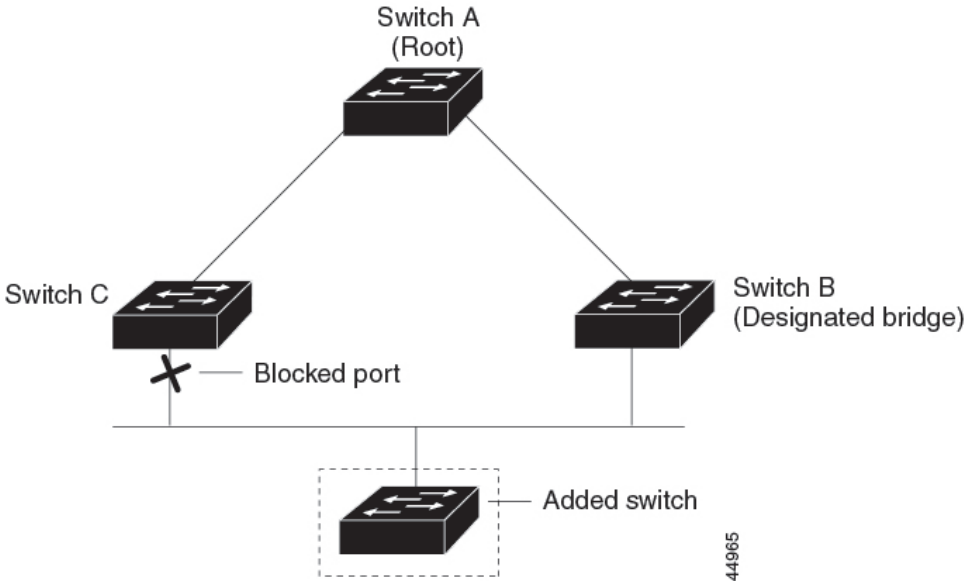


Figure 18: Adding a Switch in a Shared-Medium Topology

If a new switch is introduced into a shared-medium topology, BackboneFast is not activated because the inferior BPDUs did not come from the recognized designated switch (Switch B). The new switch begins

sending inferior BPDUs that indicate it is the root switch. However, the other switches ignore these inferior BPDUs, and the new switch learns that Switch B is the designated switch to Switch A, the root switch.



EtherChannel Guard

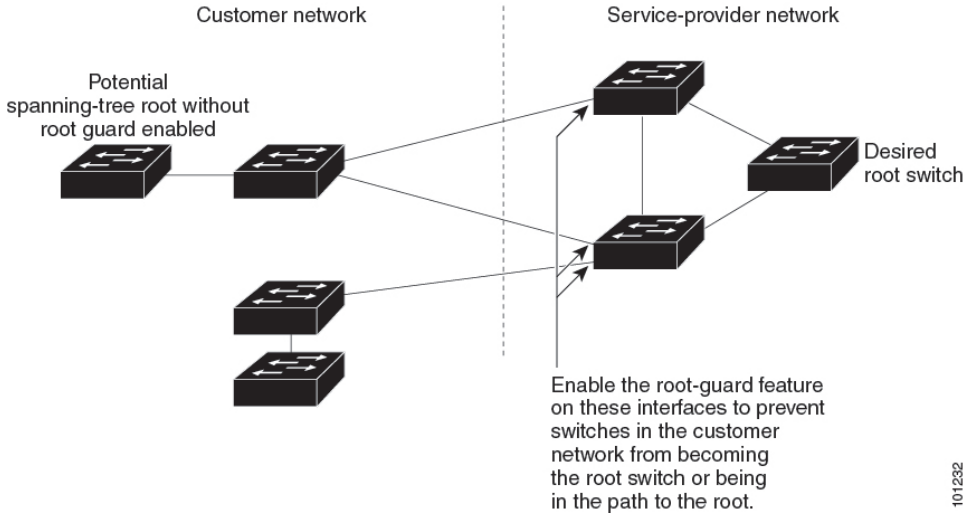
You can use EtherChannel guard to detect an EtherChannel misconfiguration between the switch and a connected device. A misconfiguration can occur if the switch interfaces are configured in an EtherChannel, but the interfaces on the other device are not. A misconfiguration can also occur if the channel parameters are not the same at both ends of the EtherChannel.

If the switch detects a misconfiguration on the other device, EtherChannel guard places the switch interfaces in the error-disabled state, and displays an error message.

Root Guard

Figure 19: Root Guard in a Service-Provider Network

The Layer 2 network of a service provider (SP) can include many connections to switches that are not owned by the SP. In such a topology, the spanning tree can reconfigure itself and select a customer switch as the root switch. You can avoid this situation by enabling root guard on SP switch interfaces that connect to switches in your customer's network. If spanning-tree calculations cause an interface in the customer network to be selected as the root port, root guard then places the interface in the root-inconsistent (blocked) state to prevent the customer's switch from becoming the root switch or being in the path to the root.



If a switch outside the SP network becomes the root switch, the interface is blocked (root-inconsistent state), and spanning tree selects a new root switch. The customer's switch does not become the root switch and is not in the path to the root.

If the switch is operating in MST mode, root guard forces the interface to be a designated port. If a boundary port is blocked in an internal spanning-tree (IST) instance because of root guard, the interface also is blocked in all MST instances. A boundary port is an interface that connects to a LAN, the designated switch of which is either an IEEE 802.1D switch or a switch with a different MST region configuration.

Root guard that is enabled on an interface applies to all the VLANs to which the interface belongs. VLANs can be grouped and mapped to an MST instance.

Caution Misuse of the root guard feature can cause a loss of connectivity.

Loop Guard

You can use loop guard to prevent alternate or root ports from becoming designated ports because of a failure that leads to a unidirectional link. This feature is most effective when it is enabled on the entire switched network. Loop guard prevents alternate and root ports from becoming designated ports, and spanning tree does not send BPDUs on root or alternate ports.

When the switch is operating in PVST+ or rapid-PVST+ mode, loop guard prevents alternate and root ports from becoming designated ports, and spanning tree does not send BPDUs on root or alternate ports.

When the switch is operating in MST mode, BPDUs are not sent on nonboundary ports only if the interface is blocked by loop guard in all MST instances. On a boundary port, loop guard blocks the interface in all MST instances.

How to Configure Optional Spanning-Tree Features

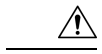
The following sections provide information about configuring Optional Spanning-Tree features:

(Optional) Enabling PortFast

An interface with the PortFast feature enabled is moved directly to the spanning-tree forwarding state without waiting for the standard forward-time delay.

If you enable the voice VLAN feature, the PortFast feature is automatically enabled. When you disable voice VLAN, the PortFast feature is not automatically disabled.

You can enable this feature if your switch is running PVST+, Rapid PVST+, or MSTP.



Caution Use PortFast only when connecting a single end station to an access or trunk port. Enabling this feature on an interface that is connected to a switch or hub could prevent spanning tree from detecting and disabling loops in your network, which could cause broadcast storms and address-learning problems.

To enable PortFast, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface to configure, and enters interface configuration mode.
Step 4	**spanning-tree portfast** [**trunk**] **Example:** `Device(config-if)# spanning-tree portfast trunk`	Enables PortFast on an access port that is connected to a single workstation or server. By specifying the **trunk** keyword, you can enable PortFast on a trunk port. **Note** To enable PortFast on trunk ports, you must use the **spanning-tree portfast trunk** interface configuration command. The **spanning-tree portfast** command will not work on trunk ports. Make sure that there are no loops in the network between the trunk port and the workstation or server before you enable PortFast on a trunk port.

	Command or Action	Purpose
		By default, PortFast is disabled on all interfaces.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

What to do next

You can use the **spanning-tree portfast default** global configuration command to globally enable the PortFast feature on all nontrunking ports.

(Optional) Enabling Bridge Protocol Data Unit Guard

You can enable the BPDU guard feature if your switch is running PVST+, Rapid PVST+, or MSTP.

Caution Configure PortFast only on ports that connect to end stations; otherwise, an accidental topology loop could cause a data packet loop and disrupt switch and network operation.

To enable BPDU guard, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree portfast bpduguard default** **Example:** `Device(config)# spanning-tree portfast bpduguard default`	Enables BPDU guard.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface that is connected to an end station, and enters interface configuration mode.
Step 5	**spanning-tree portfast** **Example:**	Enables the PortFast feature.

	Command or Action	Purpose
	`Device(config-if)# spanning-tree portfast`	
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

What to do next

To prevent the port from shutting down, you can use the **errdisable detect cause bpduguard shutdown vlan** global configuration command to shut down just the offending VLAN on the port where the violation occurred.

You also can use the **spanning-tree bpduguard enable** interface configuration command to enable BPDU guard on any port without also enabling the PortFast feature. When the port receives a BPDU, it is put it in the error-disabled state.

(Optional) Enabling Bridge Protocol Data Unit Filtering

You can also use the **spanning-tree bpdufilter enable** interface configuration command to enable BPDU filtering on any interface without also enabling the . This command prevents the interface from sending or receiving BPDUs.

Caution Enabling BPDU filtering on an interface is the same as disabling spanning tree on it and can result in spanning-tree loops.

You can enable the BPDU filtering feature if your switch is running PVST+, Rapid PVST+, or MSTP.

Caution Configure only on interfaces that connect to end stations; otherwise, an accidental topology loop could cause a data packet loop and disrupt switch and network operation.

To enable BPDU filter, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree portfast bpdufilter default**	Globally enables BPDU filtering.

	Command or Action	Purpose
	Example: `Device(config)# spanning-tree portfast bpdufilter default`	By default, BPDU filtering is disabled.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface that is connected to an end station, and enters interface configuration mode.
Step 5	**spanning-tree portfast** **Example:** `Device(config-if)# spanning-tree portfast`	Enables the PortFast feature on the specified interface.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

(Optional) Enabling UplinkFast for Use with Redundant Links

Note When you enable UplinkFast, it affects all VLANs on the switch. You cannot configure UplinkFast on an individual VLAN.

You can configure the UplinkFast feature for Rapid PVST+ or for the MSTP, but the feature remains disabled (inactive) until you change the spanning-tree mode to PVST+.

Follow these steps to enable UplinkFast and CSUF.

Before you begin

UplinkFast cannot be enabled on VLANs that have been configured with a switch priority. To enable UplinkFast on a VLAN with switch priority configured, first restore the switch priority on the VLAN to the default value using the **no spanning-tree vlan** *vlan-id* **priority** global configuration command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**spanning-tree uplinkfast** [**max-update-rate** *pkts-per-second*] **Example:** `Device(config)# spanning-tree uplinkfast max-update-rate 200`	Enables UplinkFast. (Optional) For *pkts-per-second*, the range is 0 to 32000 packets per second; the default is 150. If you set the rate to 0, station-learning frames are not generated, and the spanning-tree topology converges more slowly after a loss of connectivity. When you enter this command, CSUF also is enabled on all port interfaces.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

When UplinkFast is enabled, the switch priority of all VLANs is set to 49152. If you change the path cost to a value less than 3000 and you enable UplinkFast or UplinkFast is already enabled, the path cost of all interfaces and VLAN trunks is increased by 3000 (if you change the path cost to 3000 or above, the path cost is not altered). The changes to the switch priority and the path cost reduce the chance that a switch will become the root switch.

When UplinkFast is disabled, the switch priorities of all VLANs and path costs of all interfaces are set to default values if you did not modify them from their defaults.

When you enable the UplinkFast feature using these instructions, CSUF is automatically globally enabled on all port interfaces.

(Optional) Disabling UplinkFast

Follow these steps to disable UplinkFast .

Before you begin

UplinkFast must be enabled.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**no spanning-tree uplinkfast** **Example:** `Device(config)# no spanning-tree uplinkfast`	Disables UplinkFast and CSUF on the switch and all of its VLANs.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

When UplinkFast is disabled, the switch priorities of all VLANs and path costs of all interfaces are set to default values if you did not modify them from their defaults.

When you disable the UplinkFast feature using these instructions, CSUF is automatically globally disabled on port interfaces.

(Optional) Enabling BackboneFast

You can enable BackboneFast to detect indirect link failures and to start the spanning-tree reconfiguration sooner.

You can configure the BackboneFast feature for Rapid PVST+ or for the MSTP, but the feature remains disabled (inactive) until you change the spanning-tree mode to PVST+.

Follow these steps to enable BackboneFast on the switch.

Before you begin

If you use BackboneFast, you must enable it on all switches in the network. BackboneFast is not supported on Token Ring VLANs. This feature is supported for use with third-party switches.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree backbonefast** **Example:** `Device(config)# spanning-tree backbonefast`	Enables BackboneFast.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Enabling EtherChannel Guard

You can enable EtherChannel guard to detect an EtherChannel misconfiguration if your device is running PVST+, Rapid PVST+, or MSTP.

Follow these steps to enable EtherChannel Guard on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree etherchannel guard misconfig** **Example:** `Device(config)# spanning-tree etherchannel guard misconfig`	Enables EtherChannel guard.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

What to do next

You can use the **show interfaces status err-disabled** privileged EXEC command to show which device ports are disabled because of an EtherChannel misconfiguration. On the remote device, you can enter the **show etherchannel summary** command in privileged EXEC mode to verify the EtherChannel configuration.

After the configuration is corrected, enter the **shutdown** and **no shutdown** interface configuration commands on the port-channel interfaces that were misconfigured.

(Optional) Enabling Root Guard

Root guard that is enabled on an interface applies to all the VLANs to which the interface belongs. Do not enable the root guard on interfaces to be used by the UplinkFast feature. With UplinkFast, the backup interfaces (in the blocked state) replace the root port in the case of a failure. However, if root guard is also enabled, all

the backup interfaces used by the UplinkFast feature are placed in the root-inconsistent state (blocked) and are prevented from reaching the forwarding state.

Note You cannot enable both root guard and loop guard at the same time.

You can enable this feature if your switch is running PVST+, Rapid PVST+, or MSTP.

Follow these steps to enable root guard on the switch.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface to configure, and enters interface configuration mode.
Step 4	**spanning-tree guard root** **Example:** `Device(config-if)# spanning-tree guard root`	Enables root guard on the interface. By default, root guard is disabled on all interfaces.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

(Optional) Enabling Loop Guard

You can use loop guard to prevent alternate or root ports from becoming designated ports because of a failure that leads to a unidirectional link. This feature is most effective when it is configured on the entire switched network. Loop guard operates only on interfaces that are considered point-to-point by the spanning tree.

Note You cannot enable both loop guard and root guard at the same time.

You can enable this feature if your device is running PVST+, Rapid PVST+, or MSTP.

Follow these steps to enable loop guard on the device.

Procedure

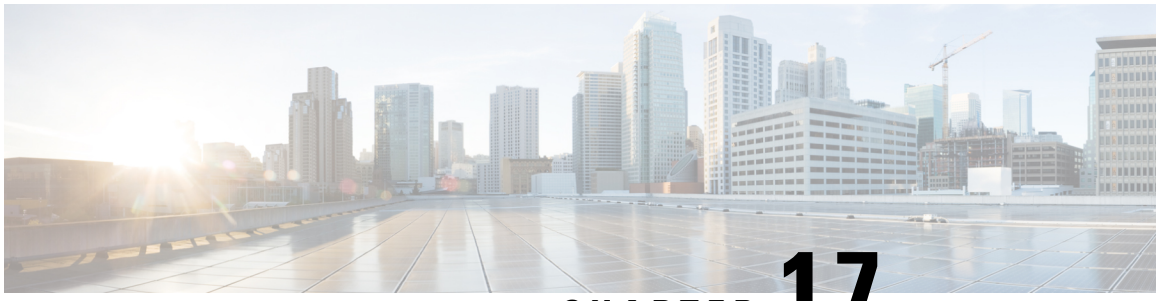
	Command or Action	Purpose
Step 1	Enter one of the following commands: • **show spanning-tree active** • **show spanning-tree mst** **Example:** `Device# show spanning-tree active` or `Device# show spanning-tree mst`	Verifies which interfaces are alternate or root ports.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree loopguard default** **Example:** `Device(config)# spanning-tree loopguard default`	Enables loop guard. By default, loop guard is disabled.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Monitoring the Spanning-Tree Status

Table 21: Commands for Monitoring the Spanning-Tree Status

Command	Purpose
show spanning-tree active	Displays spanning-tree information on active interfaces only.
show spanning-tree detail	Displays a detailed summary of interface information.
show spanning-tree interface *interface-id*	Displays spanning-tree information for the specified interface.
show spanning-tree mst interface *interface-id*	Displays MST information for the specified interface.
show spanning-tree summary [**totals**]	Displays a summary of interface states or displays the total lines of the spanning-tree state section.

Command	Purpose
show spanning-tree mst interface *interface-id* **portfast**	Displays spanning-tree PortFast information for the specified interface.



CHAPTER 17

Configuring EtherChannels

Restrictions for EtherChannels

The following are restrictions for EtherChannels:

- All ports in an EtherChannel must be assigned to the same VLAN or they must be configured as trunk port.
- The LACP 1:1 redundancy feature is supported on port channel interfaces only.

Unsupported EtherChannel features:

- You cannot configure a voice VLAN on a port channel or a member interface.
- You cannot convert an interface to an ether channel if dot1ad is configured on the interface.
- You cannot configure nonegotiate and dynamic commands on a port channel.
- You cannot configure pruning VLAN if MVRP feature is already configured on the device.
- You cannot configure network policy commands on a routed or trunk port and on an ether channel.
- You can configure the **rep segment** command only on switch port mode trunk.
- You cannot configure **switchport priority extend trust** command and **switchport priorit extend cos 3** command on an etherchannel.
- You cannot configure **platform qos low-latency** command on an interface port-channel 10.
- You cannot use Layer 2 configurations on a Layer 3 port.
- When there are any misconfigurations detected in a port mode or VLAN mask, the ports are suspended.
- On EtherChannel member port selection is software based in Layer 2 and Layer 3 multicast route. This means that all multicast traffic under a group will be routed via the same physical port of the EtherChannel.

As a result, the distribution of multicast traffic load balance over etherchannels might not evenly spread across member ports.

Information About EtherChannels

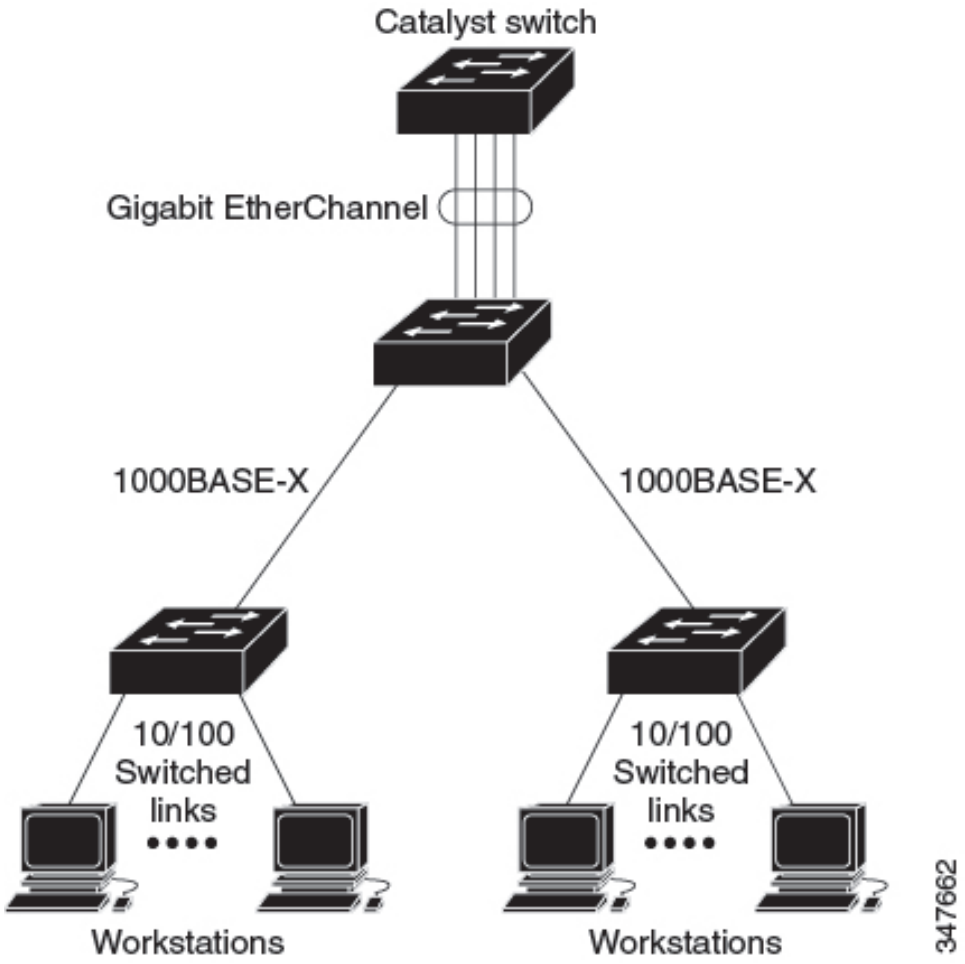
The following sections provide information about EtherChannels and the various modes to configure EtherChannels.

EtherChannel Overview

EtherChannel provides fault-tolerant high-speed links between switches, routers, and servers. You can use the EtherChannel to increase the bandwidth between the wiring closets and the data center, and you can deploy it anywhere in the network where bottlenecks are likely to occur. EtherChannel provides automatic recovery for the loss of a link by redistributing the load across the remaining links. If a link fails, EtherChannel redirects traffic from the failed link to the remaining links in the channel without intervention.

An EtherChannel consists of individual Ethernet links that are bundled into a single logical link, and each EtherChannel can consist of up to eight compatibly configured Ethernet ports.

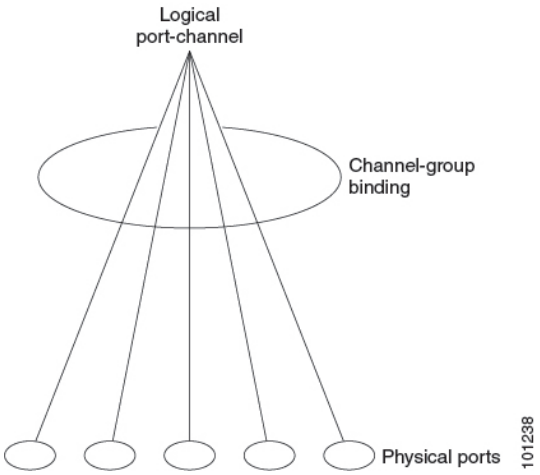
Figure 20: Typical EtherChannel Configuration



Channel Groups and Port-Channel Interfaces

An EtherChannel comprises a channel group and a port-channel interface. The channel group binds physical ports to the port-channel interface. Configuration changes applied to the port-channel interface apply to all the physical ports bound together in the channel group.

Figure 21: Relationship Between Physical Ports, a Channel Group, and a Port-Channel Interface



The **channel-group** command binds the physical port and the port-channel interface together. Each EtherChannel has a port-channel logical interface numbered from 1 to 48. This port-channel interface number corresponds to the one specified with the **channel-group** interface configuration command.

- With Layer 2 ports, use the **channel-group** interface configuration command to dynamically create the port-channel interface.

  You also can use the **interface port-channel** *port-channel-number* global configuration command to manually create the port-channel interface, but then you must use the **channel-group** *channel-group-number* command to bind the logical interface to a physical port. The *channel-group-number* can be the same as the *port-channel-number,* or you can use a new number. If you use a new number, the **channel-group** command dynamically creates a new port channel.

- With Layer 3 ports, you should manually create the logical interface by using the **interface port-channel** global configuration command followed by the **no switchport** interface configuration command. You then manually assign an interface to the EtherChannel by using the **channel-group** interface configuration command.

Port Aggregation Protocol

The Port Aggregation Protocol (PAgP) is a Cisco-proprietary protocol that can be run only on Cisco devices and on those devices that are licensed by vendors to support PAgP. PAgP facilitates the automatic creation of EtherChannels by exchanging PAgP packets between Ethernet ports.

By using PAgP, the switch learns the identity of partners capable of supporting PAgP and the capabilities of each port. It then dynamically groups similarly configured ports into a single logical link (channel or aggregate port). Similarly configured ports are grouped based on hardware, administrative, and port parameter constraints. For example, PAgP groups the ports with the same speed, duplex mode, native VLAN, VLAN range, and trunking status and type. After grouping the links into an EtherChannel, PAgP adds the group to the spanning tree as a single device port.

Port Aggregation Protocol Modes

PAgP modes specify whether a port can send PAgP packets, which start PAgP negotiations, or only respond to PAgP packets received.

Table 22: EtherChannel PAgP Modes

Mode	Description
auto	Places a port into a passive negotiating state, in which the port responds to PAgP packets it receives but does not start PAgP packet negotiation. This setting minimizes the transmission of PAgP packets.
desirable	Places a port into an active negotiating state, in which the port starts negotiations with other ports by sending PAgP packets.

Switch ports exchange PAgP packets only with partner ports that are configured in the **auto** or **desirable** modes. Ports that are configured in the **on** mode do not exchange PAgP packets.

Both the **auto** and **desirable** modes enable ports to negotiate with partner ports to form an EtherChannel based on criteria such as port speed. and for Layer 2 EtherChannels, based on trunk state and VLAN numbers.

Ports can form an EtherChannel when they are in different PAgP modes as long as the modes are compatible. For example:

- A port in the **desirable** mode can form an EtherChannel with another port that is in the **desirable** or **auto** mode.
- A port in the **auto** mode can form an EtherChannel with another port in the **desirable** mode.

A port in the **auto** mode cannot form an EtherChannel with another port that is also in the **auto** mode because neither port starts PAgP negotiation.

Silent Mode

If your switch is connected to a partner that is PAgP-capable, you can configure the switch port for nonsilent operation by using the **non-silent** keyword. If you do not specify **non-silent** with the **auto** or **desirable** mode, silent mode is assumed.

Use the silent mode when the switch is connected to a device that is not PAgP-capable and seldom, if ever, sends packets. An example of a silent partner is a file server or a packet analyzer that is not generating traffic. In this case, running PAgP on a physical port that is connected to a silent partner prevents that switch port from ever becoming operational. However, the silent setting allows PAgP to operate, to attach the port to a channel group, and to use the port for transmission.

Port Aggregation Protocol Learn Method and Priority

Network devices are classified as PAgP physical learners or aggregate-port learners. A device is a physical learner if it learns addresses by physical ports and directs transmissions based on that knowledge. A device is an aggregate-port learner if it learns addresses by aggregate (logical) ports. The learn method must be configured the same at both ends of the link.

When a device and its partner are both aggregate-port learners, they learn the address on the logical port-channel. The device sends packets to the source by using any of the ports in the EtherChannel. With aggregate-port learning, it is not important on which physical port the packet arrives.

PAgP cannot automatically detect when the partner device is a physical learner and when the local device is an aggregate-port learner. Therefore, you must manually set the learning method on the local device to learn addresses by physical ports. You also must set the load-distribution method to source-based distribution, so that any given source MAC address is always sent on the same physical port.

You also can configure a single port within the group for all transmissions and use other ports for hot-standby. The unused ports in the group can be swapped into operation in just a few seconds if the selected single port loses hardware-signal detection. You can configure which port is always selected for packet transmission by changing its priority with the **pagp port-priority** interface configuration command. The higher the priority, the more likely that the port will be selected.

Note The device supports address learning only on aggregate ports even though the **physical-port** keyword is provided in the CLI. The **pagp learn-method** command and the **pagp port-priority** command have no effect on the device hardware, but they are required for PAgP interoperability with devices that only support address learning by physical ports.

When the link partner of the device is a physical learner, we recommend that you configure the device as a physical-port learner by using the **pagp learn-method physical-port** interface configuration command. Set the load-distribution method based on the source MAC address by using the **port-channel load-balance src-mac** global configuration command. The device then sends packets to the physical learner using the same port in the EtherChannel from which it learned the source address. Only use the **pagp learn-method** command in this situation.

Port Aggregation Protocol Interaction with Other Features

The Dynamic Trunking Protocol (DTP) and the Cisco Discovery Protocol (CDP) send and receive packets over the physical ports in the EtherChannel. Trunk ports send and receive PAgP protocol data units (PDUs) on the lowest numbered VLAN.

In Layer 2 EtherChannels, the first port in the channel that comes up provides its MAC address to the EtherChannel. If this port is removed from the bundle, one of the remaining ports in the bundle provides its MAC address to the EtherChannel. For Layer 3 EtherChannels, the MAC address is allocated by the active device as soon as the interface is created (through the **interface port-channel** global configuration command).

PAgP sends and receives PAgP PDUs only from ports that are up and have PAgP enabled for the auto or desirable mode.

Link Aggregation Control Protocol

The LACP is defined in IEEE 802.3ad and enables Cisco devices to manage Ethernet channels between devices that conform to the IEEE 802.3ad protocol. LACP facilitates the automatic creation of EtherChannels by exchanging LACP packets between Ethernet ports.

By using LACP, the switch learns the identity of partners capable of supporting LACP and the capabilities of each port. It then dynamically groups similarly configured ports into a single logical link (channel or aggregate port). Similarly configured ports are grouped based on hardware, administrative, and port parameter constraints. For example, LACP groups the ports with the same speed, duplex mode, native VLAN, VLAN range, and trunking status and type. After grouping the links into an EtherChannel, LACP adds the group to the spanning tree as a single device port.

The independent mode behavior of ports in a port channel is changed. By default, standalone mode is enabled. When no response is received from an LACP peer, ports in the port channel are moved to suspended state.

Link Aggregation Control Protocol Modes

LACP modes specify whether a port can send LACP packets or only receive LACP packets.

Table 23: EtherChannel LACP Modes

Mode	Description
active	Places a port into an active negotiating state in which the port starts negotiations with other ports by sending LACP packets.
passive	Places a port into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation. This setting minimizes the transmission of LACP packets.

Both the **active** and **passive LACP** modes enable ports to negotiate with partner ports to an EtherChannel based on criteria such as port speed, and for Layer 2 EtherChannels, based on trunk state and VLAN numbers.

Ports can form an EtherChannel when they are in different LACP modes as long as the modes are compatible. For example:

- A port in the **active** mode can form an EtherChannel with another port that is in the **active** or **passive** mode.
- A port in the **passive** mode cannot form an EtherChannel with another port that is also in the **passive** mode because neither port starts LACP negotiation.

Link Aggregation Control Protocol Standalone Mode on Ethernet Channel

When one end of an EtherChannel has more members than the other, the unmatched ports enter the standalone state. The standalone mode is also called the independent mode. In the standalone mode the port is not bundled in an EtherChannel. The port functions as a standalone data port and it can send and receive BPDUs and data traffic.

In a topology that is not protected from Layer 2 loops by the spanning tree protocol (STP), a port in the standalone state can cause significant network errors. You can enter the **port-channel standalone-disable** command in the interface configuration mode to put ports into the suspended state instead of the standalone state.

The standalone mode is particularly relevant when a port (A) in a Layer 2 LACP EtherChannel is connected to an unresponsive port (B) on the peer. When LACP standalone is disabled on the EtherChannel, all traffic arriving on A is blocked (the default behavior on a switch). In some scenarios, you might want to allow management traffic on such ports. You can do this by enabling LACP standalone (or independent) mode. To enable the standalone mode on a Layer 2 LACP Etherchannel, use the **no port-channel standalone disable** command in the interface configuration mode. To disable the Standalone mode and revert to the default use the **port-channel standalone disable** command in the interface configuration mode.

Note LACP standalone mode is disabled by default.

You can configure the LACP standalone mode on a Layer 3 EtherChannel. To configure the standalone mode use the **no port-channel standalone disable** command in the interface configuration mode. To disable the Standalone mode and revert to the default use the **port-channel standalone disable** command in the interface configuration mode.

Link Aggregation Control Protocol and Link Redundancy

LACP port-channel operation, bandwidth availability, and link redundancy can be further refined with the LACP port-channel min-links and the LACP max-bundle features.

The LACP port-channel min-links feature:

- Configures the minimum number of ports that must be linked up and bundled in the LACP port channel.
- Prevents a low-bandwidth LACP port channel from becoming active.
- Causes an LACP port channel to become inactive if there are too few active members ports to supply the required minimum bandwidth.

The LACP max-bundle feature:

- Defines an upper limit on the number of bundled ports in an LACP port channel.
- Allows hot-standby ports with fewer bundled ports. For example, in an LACP port channel with five ports, you can specify a max-bundle of three, and the two remaining ports are designated as hot-standby ports.

Link Aggregation Control Protocol Interaction with Other Features

The DTP and the CDP send and receive packets over the physical ports in the EtherChannel. Trunk ports send and receive LACP PDUs on the lowest numbered VLAN.

In Layer 2 EtherChannels, the first port in the channel that comes up provides its MAC address to the EtherChannel. If this port is removed from the bundle, one of the remaining ports in the bundle provides its MAC address to the EtherChannel. For Layer 3 EtherChannels, the MAC address is allocated by the active device as soon as the interface is created through the **interface port-channel** global configuration command.

LACP sends and receives LACP PDUs only from ports that are up and have LACP enabled for the active or passive mode.

Link Aggregation Control Protocol Interaction with Other Features 1:1 Redundancy

The LACP 1:1 Redundancy feature supports an EtherChannel configuration with one active link, and fast switchover to a hot-standby link. The link that is connected to the port with the lower port priority number (and therefore, of a higher priority) will be the active link, and the other link will be in a hot-standby state. If the active link goes down, LACP performs a fast switchover to the hot-standby link to keep the EtherChannel up. When the failed link becomes operational again, LACP performs another fast switchover to revert to the original active link.

To allow the higher priority port to stabilize when it becomes active again after a higher-priority to lower-priority switchover, the LACP 1:1 Hot Standby Dampening feature configures a timer that delays switchover back to the higher priority port after higher priority port becomes active.

EtherChannel On Mode

EtherChannel **on** mode can be used to manually configure an EtherChannel. The **on** mode forces a port to join an EtherChannel without negotiations. The **on** mode can be useful if the remote device does not support PAgP or LACP. In the **on** mode, a usable EtherChannel exists only when the devices at both ends of the link are configured in the **on** mode.

Ports that are configured in the **on** mode in the same channel group must have compatible port characteristics, such as speed and duplex. Ports that are not compatible are suspended, even though they are configured in the **on** mode.

Caution You should use care when using the **on** mode. This is a manual configuration, and ports on both ends of the EtherChannel must have the same configuration. If the group is misconfigured, packet loss or spanning-tree loops can occur.

Load-Balancing and Forwarding Methods

EtherChannel balances the traffic load across the links in a channel by reducing part of the binary pattern that is formed from the addresses in the frame to a numerical value that selects one of the links in the channel. You can specify one of several different load-balancing modes, including load distribution based on MAC addresses, IP addresses, source addresses, destination addresses, or both source and destination addresses. The selected mode applies to all EtherChannels configured on the device.

Note Layer 3 Equal-cost multi path (ECMP) load balancing is based on source IP address, destination IP address, source port, destination port, and layer 4 protocol. Fragmented packets will be treated on two different links based on the algorithm that is calculated using these parameters. Any changes in one of these parameters result in load balancing.

MAC Address Forwarding

With source-MAC address forwarding, when packets are forwarded to an EtherChannel, they are distributed across the ports in the channel based on the source-MAC address of the incoming packet. Therefore, to provide load-balancing, packets from different hosts use different ports in the channel, but packets from the same host use the same port in the channel.

With destination-MAC address forwarding, when packets are forwarded to an EtherChannel, they are distributed across the ports in the channel based on the destination host's MAC address of the incoming packet. Therefore, packets to the same destination are forwarded over the same port, and packets to a different destination are sent on a different port in the channel.

With source-and-destination MAC address forwarding, when packets are forwarded to an EtherChannel, they are distributed across the ports in the channel based on both the source and destination MAC addresses. This forwarding method, a combination source-MAC and destination-MAC address forwarding methods of load distribution, can be used if it is not clear whether source-MAC or destination-MAC address forwarding is better suited on a particular device. With source-and-destination MAC-address forwarding, packets sent from host A to host B, host A to host C, and host C to host B could all use different ports in the channel.

IP Address Forwarding

With source-IP address-based forwarding, packets are distributed across the ports in the EtherChannel based on the source-IP address of the incoming packet. To provide load balancing, packets from different IP addresses use different ports in the channel, and packets from the same IP address use the same port in the channel.

With destination-IP address-based forwarding, packets are distributed across the ports in the EtherChannel based on the destination-IP address of the incoming packet. To provide load balancing, packets from the same IP source address that is sent to different IP destination addresses could be sent on different ports in the channel. Packets sent from different source IP addresses to the same destination IP address are always sent on the same port in the channel.

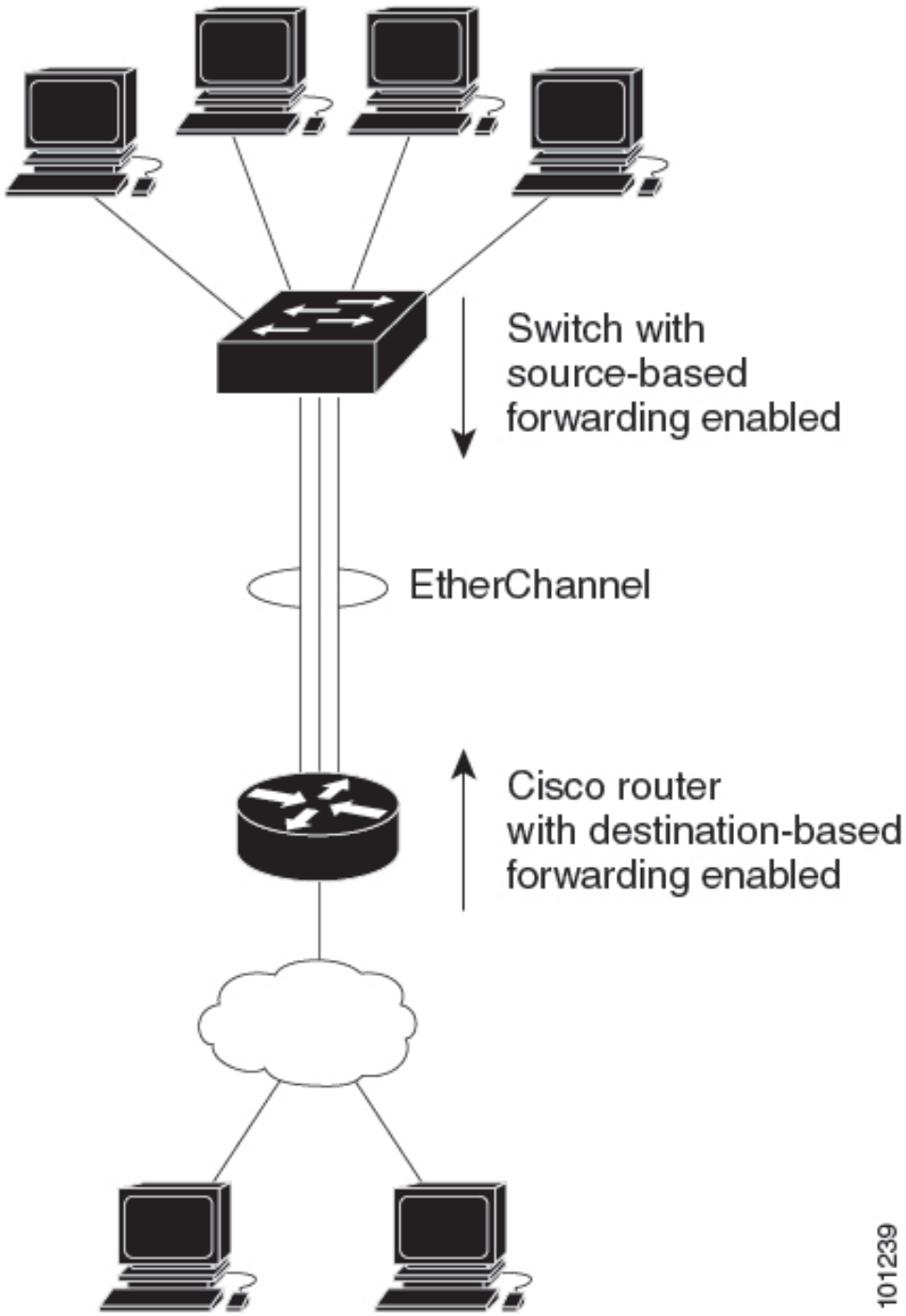
With source-and-destination IP address-based forwarding, packets are distributed across the ports in the EtherChannel based on both the source and destination IP addresses of the incoming packet. This forwarding method, a combination of source-IP and destination-IP address-based forwarding, can be used if it is not clear whether source-IP or destination-IP address-based forwarding is better suited on a particular device. In this method, packets sent from the IP address A to IP address B, from IP address A to IP address C, and from IP address C to IP address B could all use different ports in the channel.

Load-Balancing Advantages

Different load-balancing methods have different advantages, and the choice of a particular load-balancing method should be based on the position of the device in the network and the kind of traffic that needs to be load-distributed.

Figure 22: Load Distribution and Forwarding Methods

In the following figure, an EtherChannel of four workstations communicates with a router. Because the router is a single MAC-address device, source-based forwarding on the switch EtherChannel ensures that the switch uses all available bandwidth to the router. The router is configured for destination-based forwarding because the large number of workstations ensures that the traffic is evenly distributed from the router EtherChannel.



Use the option that provides the greatest variety in your configuration. For example, if the traffic on a channel is going only to a single MAC address, using the destination-MAC address always chooses the same link in the channel. Using source addresses or IP addresses might result in better load-balancing.

Default EtherChannel Configuration

The default EtherChannel configuration is described in this table.

Table 24: Default EtherChannel Configuration

Feature	Default Setting
Channel groups	None assigned.
Port-channel logical interface	None defined.
PAgP mode	No default.
PAgP learn method	Aggregate-port learning on all ports.
PAgP priority	128 on all ports.
LACP mode	No default.
LACP learn method	Aggregate-port learning on all ports.
LACP port priority	32768 on all ports.
LACP system priority	32768.
LACP system ID	LACP system priority and MAC address.
Load-balancing	Load distribution on the switch is based on the source-MAC address of the incoming packet. The source-MAC address is **src-mac**.

EtherChannel Configuration Guidelines

If improperly configured, some EtherChannel ports are automatically disabled to avoid network loops and other problems. Follow these guidelines to avoid configuration problems:

- A maximum of 48 EtherChannels are supported on a switch.
- Configure all ports in an EtherChannel to operate at the same speeds and duplex modes.
- Enable all ports in an EtherChannel. A port in an EtherChannel that is disabled by using the **shutdown** interface configuration command is treated as a link failure, and its traffic is transferred to one of the remaining ports in the EtherChannel.
- When a group is first created, all ports follow the parameters set for the first port to be added to the group. If you change the configuration of one of these parameters, you must also make the changes to all ports in the group:
  - Allowed-VLAN list
  - Spanning-tree path cost for each VLAN
  - Spanning-tree port priority for each VLAN

    - Spanning-tree Port Fast setting

- Do not configure a port to be a member of more than one EtherChannel group.
- Do not configure an EtherChannel in both the PAgP and LACP modes. EtherChannel groups running PAgP and LACP can coexist on the same switch. Individual EtherChannel groups can run either PAgP or LACP, but they cannot interoperate.
- Do not configure a port that is an active or a not-yet-active member of an EtherChannel as an IEEE 802.1x port. If you try to enable IEEE 802.1x on an EtherChannel port, an error message appears, and IEEE 802.1x is not enabled.
- If EtherChannels are configured on device interfaces, remove the EtherChannel configuration from the interfaces before globally enabling IEEE 802.1x on a device by using the **dot1x system-auth-control** global configuration command.

Layer 2 EtherChannel Configuration Guidelines

When configuring Layer 2 EtherChannels, follow these guidelines:

- Assign all ports in the EtherChannel to the same VLAN, or configure them as trunks. Ports with different native VLANs cannot form an EtherChannel.
- An EtherChannel supports the same allowed range of VLANs on all the ports in a trunking Layer 2 EtherChannel. If the allowed range of VLANs is not the same, the ports do not form an EtherChannel even when PAgP is set to the **auto** or **desirable** mode.
- Ports with different spanning-tree path costs can form an EtherChannel if they are otherwise compatibly configured. Setting different spanning-tree path costs does not, by itself, make ports incompatible for the formation of an EtherChannel.

Layer 3 EtherChannel Configuration Guidelines

For Layer 3 EtherChannels, assign the Layer 3 address to the port-channel logical interface, not to the physical ports in the channel.

Auto-LAG

The auto-LAG feature provides the ability to auto create EtherChannels on ports that are connected to a switch. By default, auto-LAG is disabled globally and is enabled on all port interfaces. The auto-LAG applies to a switch only when it is enabled globally.

On enabling auto-LAG globally, the following scenarios are possible:

- All port interfaces participate in creation of auto EtherChannels provided the partner port interfaces have EtherChannel configured on them. For more information, see the *"The supported auto-LAG configurations between the actor and partner devices"* table below.
- Ports that are already part of manual EtherChannels cannot participate in creation of auto EtherChannels.
- When auto-LAG is disabled on a port interface that is already a part of an auto created EtherChannel, the port interface unbundles from the auto EtherChannel.

The following table shows the supported auto-LAG configurations between the actor and partner devices:

Table 25: The supported auto-LAG configurations between the actor and partner devices

Actor/Partner	Active	Passive	Auto
Active	Yes	Yes	Yes
Passive	Yes	No	Yes
Auto	Yes	Yes	Yes

On disabling auto-LAG globally, all auto created Etherchannels become manual EtherChannels.

You cannot add any configurations in an existing auto created EtherChannel. To add, you should first convert it into a manual EtherChannel by executing the **port-channel**<*channel-number*>**persistent**.

Note Auto-LAG uses the LACP protocol to create auto EtherChannel. Only one EtherChannel can be automatically created with the unique partner devices.

Auto-LAG Configuration Guidelines

Follow these guidelines when configuring the auto-LAG feature.

- When auto-LAG is enabled globally and on the port interface, and if you do not want the port interface to become a member of the auto EtherChannel, disable the auto-LAG on the port interface.
- A port interface will not bundle to an auto EtherChannel when it is already a member of a manual EtherChannel. To allow it to bundle with the auto EtherChannel, first unbundle the manual EtherChannel on the port interface.
- When auto-LAG is enabled and auto EtherChannel is created, you can create multiple EtherChannels manually with the same partner device. But by default, the port tries to create auto EtherChannel with the partner device.
- The auto-LAG is supported only on Layer 2 EtherChannel. It is not supported on Layer 3 interface and Layer 3 EtherChannel.

How to Configure EtherChannels

After you configure an EtherChannel, configuration changes applied to the port-channel interface apply to all the physical ports assigned to the port-channel interface, and configuration changes that are applied to the physical port affect only the port where you apply the configuration.

The following sections provide various configuration information for EtherChannels:

Configuring Layer 2 EtherChannels

Configure Layer 2 EtherChannels by assigning ports to a channel group with the **channel-group** command in interface configuration mode. This command automatically creates the port-channel logical interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies a physical port, and enters interface configuration mode. Valid interfaces are physical ports. For a PAgP EtherChannel, you can configure up to eight ports of the same type and speed for the same group. For a LACP EtherChannel, you can configure up to 16 Ethernet ports of the same type. Up to eight ports can be active, and up to eight ports can be in standby mode.
Step 4	**switchport mode** {**access** | **trunk**} **Example:** `Device(config-if)# switchport mode access`	Assigns all ports as static-access ports in the same VLAN, or configure them as trunks. If you configure the port as a static-access port, assign it to only one VLAN. The range is 1 to 4094.
Step 5	**switchport access vlan** *vlan-id* **Example:** `Device(config-if)# switchport access vlan 22`	(Optional) If you configure the port as a static-access port, assign it to only one VLAN. The range is 1 to 4094.
Step 6	**channel-group** *channel-group-number* **mode** {**auto** [**non-silent**] | **desirable** [**non-silent**] | **on** } | { **active** | **passive**} **Example:** `Device(config-if)# channel-group 5 mode auto`	Assigns the port to a channel group, and specifies the PAgP or the LACP mode. For **mode**, select one of these keywords: • **auto** –Enables PAgP only if a PAgP device is detected. It places the port into a passive negotiating state, in which the port responds to PAgP packets it receives but does not start PAgP packet negotiation. • **desirable** –Unconditionally enables PAgP. It places the port into an active negotiating state, in which the port starts negotiations with other ports by sending PAgP packets.

	Command or Action	Purpose
		• **on** –Forces the port to channel without PAgP or LACP. In the **on** mode, an EtherChannel exists only when a port group in the **on** mode is connected to another port group in the **on** mode. • **non-silent** –(Optional) If your device is connected to a partner that is PAgP-capable, configures the device port for nonsilent operation when the port is in the **auto** or **desirable** mode. If you do not specify **non-silent**, silent is assumed. The silent setting is for connections to file servers or packet analyzers. This setting allows PAgP to operate, to attach the port to a channel group, and to use the port for transmission. • **active**—Enables LACP only if a LACP device is detected. It places the port into an active negotiating state in which the port starts negotiations with other ports by sending LACP packets. • **passive** –Enables LACP on the port and places it into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation.
Step 7	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring Layer 3 EtherChannels

Follow these steps to assign an Ethernet port to a Layer 3 EtherChannel. This procedure is required.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies a physical port, and enters interface configuration mode. Valid interfaces include physical ports. For a PAgP EtherChannel, you can configure up to eight ports of the same type and speed for the same group. For a LACP EtherChannel, you can configure up to 16 Ethernet ports of the same type. Up to eight ports can be active, and up to eight ports can be in standby mode.
Step 4	**no ip address** **Example:** `Device(config-if)# no ip address`	Ensures that there is no IP address assigned to the physical port.
Step 5	**no switchport** **Example:** `Device(config-if)# no switchport`	Puts the port into Layer 3 mode.
Step 6	**channel-group** *channel-group-number* **mode** { **auto** [**non-silent**] | **desirable** [**non-silent**] | **on** } | { **active** | **passive** } **Example:** `Device(config-if)# channel-group 5 mode auto`	Assigns the port to a channel group, and specifies the PAgP or the LACP mode. For **mode**, select one of these keywords: • **auto**—Enables PAgP only if a PAgP device is detected. It places the port into a passive negotiating state, in which the port responds to PAgP packets it receives but does not start PAgP packet negotiation. • **desirable**—Unconditionally enables PAgP. It places the port into an active negotiating state, in which the port starts negotiations with other ports by sending PAgP packets. • **on**—Forces the port to channel without PAgP or LACP. In the **on** mode, an EtherChannel exists only when a port group in the **on** mode is connected to another port group in the **on** mode. • **non-silent**—(Optional) If your device is connected to a partner that is PAgP capable, configures the device port for

	Command or Action	Purpose
		nonsilent operation when the port is in the **auto** or **desirable** mode. If you do not specify **non-silent**, silent is assumed. The silent setting is for connections to file servers or packet analyzers. This setting allows PAgP to operate, to attach the port to a channel group, and to use the port for transmission. • **active**—Enables LACP only if a LACP device is detected. It places the port into an active negotiating state in which the port starts negotiations with other ports by sending LACP packets. • **passive** –Enables LACP on the port and places it into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation.
Step 7	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

(Optional) Configuring EtherChannel Load-Balancing

You can configure EtherChannel load-balancing to use one of several different forwarding methods.

To configure EtherChannel Load-balancing, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**port-channel load-balance** {**dst-ip** | **dst-mac** | **dst-mixed-ip-port** | **dst-port** | **extended** | **src-dst-ip** | **src-dst-mac** | **src-dst-mixed-ip-port** | **src-dst-port**	Configures an EtherChannel load-balancing method. The default is **src-mac**. Select one of these load-distribution methods:

	Command or Action	Purpose
	| **src-ip** | **src-mac** | **src-mixed-ip-port** | **src-port** } **Example:** `Device(config)# port-channel load-balance src-mac`	• **dst-ip**—Specifies destination-host IP address. • **dst-mac**—Specifies the destination-host MAC address of the incoming packet. • **dst-mixed-ip-port**—Specifies the host IP address and TCP/UDP port. • **dst-port**—Specifies the destination TCP/UDP port. • **src-dst-ip**—Specifies the source and destination host IP address. • **src-dst-mac**—Specifies the source and destination host MAC address. • **src-dst-mixed-ip-port**—Specifies the source and destination host IP address and TCP/UDP port. • **src-dst-port**—Specifies the source and destination TCP/UDP port. • **extended**—Specifies extended load balance methods--combinations of source and destination methods beyond those available with the standard command. • **src-ip**—Specifies the source host IP address. • **src-mac**—Specifies the source MAC address of the incoming packet. • **src-mixed-ip-port**—Specifies the source host IP address and TCP/UDP port. • **src-port**—Specifies the source TCP/UDP port.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring EtherChannel Extended Load-Balancing

Configure EtherChannel extended load-balancing when you want to use a combination of load-balancing methods.

To configure EtherChannel extended load-balancing, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**port-channel load-balance extended** {**dst-ip** | **dst-mac dst-port** | **ipv6-label** | **l3-proto** | **src-ip** | **src-mac** | **src-port** } **Example:** `Device(config)# port-channel load-balance extended dst-ip dst-mac src-ip`	Configures an EtherChannel extended load-balancing method. The default is **src-mac**. Select one of these load-distribution methods: • **dst-ip**—Specifies destination-host IP address. • **dst-mac**—Specifies the destination-host MAC address of the incoming packet. • **dst-port**—Specifies the destination TCP/UDP port. • **ipv6-label**—Specifies the IPv6 flow label. • **l3-proto**—Specifies the Layer 3 protocol. • **src-ip**—Specifies the source host IP address. • **src-mac**—Specifies the source MAC address of the incoming packet. • **src-port**—Specifies the source TCP/UDP port.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Port Aggregation Protocol Learn Method and Priority

To configure the PAgP learn method and priority, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port for transmission, and enters interface configuration mode.
Step 4	**pagp learn-method physical-port** **Example:** `Device(config-if)# pagp learn-method physical port`	Selects the PAgP learning method. By default, **aggregation-port learning** is selected, which means the device sends packets to the source by using any of the ports in the EtherChannel. With aggregate-port learning, it is not important on which physical port the packet arrives. Selects **physical-port** to connect with another device that `is` a physical learner. Make sure to configure the **port-channel load-balance** global configuration command to **src-mac**. The learning method must be configured the same at both ends of the link.
Step 5	**pagp port-priority** *priority* **Example:** `Device(config-if)# pagp port-priority 200`	Assigns a priority so that the selected port is chosen for packet transmission. For *priority*, the range is 0 to 255. The default is 128. The higher the priority, the more likely that the port will be used for PAgP transmission.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring Link Aggregation Control Protocol Hot-Standby Ports

When LACP is enabled, the software, by default, tries to configure the maximum number of LACP-compatible ports in a channel, up to a maximum of 16 ports. Only eight LACP links can be active at one time; the remaining

eight links are placed in hot-standby mode. If one of the active links becomes inactive, a link that is in the hot-standby mode becomes active in its place.

You can override the default behavior by specifying the maximum number of active ports in a channel, in which case, the remaining ports become hot-standby ports. For example, if you specify a maximum of five ports in a channel, up to 11 ports become hot-standby ports.

If you configure more than eight links for an EtherChannel group, the software automatically decides which of the hot-standby ports to make active based on the LACP priority. To every link between systems that operate LACP, the software assigns a unique priority that is made up of these elements (in priority order):

- LACP system priority
- System ID (the device MAC address)
- LACP port priority
- Port number

In priority comparisons, numerically lower values have higher priority. The priority decides which ports should be put in standby mode when there is a hardware limitation that prevents all compatible ports from aggregating.

Determining which ports are active and which are hot standby is a two-step procedure. First the system with a numerically lower system priority and system ID is placed in charge of the decision. Next, that system decides which ports are active and which are hot standby, based on its values for port priority and port number. The port priority and port number values for the other system are not used.

You can change the default values of the LACP system priority and the LACP port priority to affect how the software selects active and standby links.

(Optional) Configuring the Link Aggregation Control Protocol Max Bundle

When you specify the maximum number of bundled LACP ports allowed in a port channel, the remaining ports in the port channel are designated as hot-standby ports.

Beginning in privileged EXEC mode, follow these steps to configure the maximum number of LACP ports in a port channel.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-number* **Example:**	Enters interface configuration mode for a port channel. For *channel-number*, the range is 1 to 48.

	Command or Action	Purpose
	`Device(config)# interface port-channel 2`	
Step 4	**lacp max-bundle** *max-bundle-number* **Example:** `Device(config-if)# lacp max-bundle 3`	Specifies the maximum number of LACP ports in the port-channel bundle. The range is 1 to 8.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring Link Aggregation Control Protocol Port-Channel Standalone Disable

To disable the standalone EtherChannel member port state on a port channel, perform this task on the port channel interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-group* **Example:** `Device(config)# interface port-channel channel-group`	Selects a port channel interface to configure.
Step 4	**port-channel standalone-disable** **Example:** `Device(config-if)# port-channel standalone-disable`	Disables the standalone mode on the port-channel interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits configuration mode.
Step 6	**show etherchannel** **Example:** `Device# show etherchannel channel-group`	Verifies the configuration.

	Command or Action	Purpose
	port-channel Device# **show etherchannel** *channel-group* **detail**	

Configuring Link Aggregation Control Protocol Standalone Mode on EtherChannel

To configure LACP Standalone or Independent mode on an EtherChannel, perform the following procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** Device> **enable**	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** Device# **configure terminal**	Enters global configuration mode.
Step 3	**interface port-channel** *channel-number* **Example:** Device(config)# **interface port-channel 1**	Enters interface configuration mode for a port channel.
Step 4	**no port-channel standalone-disable** **Example:** Device(config-if)#**no port-channel standalone-disable**	Enables the LACP standalone or independent mode.
Step 5	**end** **Example:** Device(config-if)#**end**	Returns to privileged EXEC mode.

Configuring the Link Aggregation Control Protocol Port Channel Min-Links

You can specify the minimum number of active ports that must be in the link-up state and bundled in an EtherChannel for the port channel interface to transition to the link-up state. Using EtherChannel min-links, you can prevent low-bandwidth LACP EtherChannels from becoming active. Port channel min-links also cause LACP EtherChannels to become inactive if they have too few active member ports to supply the required minimum bandwidth.

To configure the minimum number of links that are required for a port channel. Perform the following tasks.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-number* **Example:** `Device(config)# interface port-channel 2`	Enters interface configuration mode for a port-channel. For *channel-number*, the range is 1 to 48.
Step 4	**port-channel min-links** *min-links-number* **Example:** `Device(config-if)# port-channel min-links 3`	Specifies the minimum number of member ports that must be in the link-up state and bundled in the EtherChannel for the port channel interface to transition to the link-up state. For *min-links-number*, the range is 2 to 8.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Link Aggregation Control Protocol System Priority

You can configure the system priority for all the EtherChannels that are enabled for LACP by using the **lacp system-priority** command in global configuration mode. You cannot configure a system priority for each LACP-configured channel. By changing this value from the default, you can affect how the software selects active and standby links.

You can use the **show etherchannel summary** command in privileged EXEC mode to see which ports are in the hot-standby mode (denoted with an H port-state flag).

Follow these steps to configure the LACP system priority.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**lacp system-priority** *priority* **Example:** `Device(config)# lacp system-priority 32000`	Configures the LACP system priority. The range is 1 to 65535. The default is 32768. The lower the value, the higher the system priority.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

(Optional) Configuring the Link Aggregation Control Protocol Port Priority

By default, all ports use the same port priority. If the local system has a lower value for the system priority and the system ID than the remote system, you can affect which of the hot-standby links become active first by changing the port priority of LACP EtherChannel ports to a lower value than the default. The hot-standby ports that have lower port numbers become active in the channel first. You can use the **show etherchannel summary** privileged EXEC command to see which ports are in the hot-standby mode (denoted with an H port-state flag).

Note If LACP is not able to aggregate all the ports that are compatible (for example, the remote system might have more restrictive hardware limitations), all the ports that cannot be actively included in the EtherChannel are put in the hot-standby state and are used only if one of the channeled ports fails.

Follow these steps to configure the LACP port priority.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:**	Specifies the port to be configured, and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet 1/1`	
Step 4	**lacp port-priority** *priority* **Example:** `Device(config-if)# lacp port-priority 32000`	Configures the LACP port priority. The range is 1 to 65535. The default is 32768. The lower the value, the more likely that the port will be used for LACP transmission.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring Link Aggregation Control Protocol 1:1 Redundancy

Note

- LACP 1:1 redundancy must be enabled at both ends of the LACP EtherChannel.
- For the LACP 1:1 Redundancy feature to work, the **lacp max-bundle 1** command must be configured along with the **lacp fast-switchover** command.
- For the LACP 1:1 Hot Standby Dampening feature to work, the **lacp max-bundle 1** and **lacp fast-switchover** commands must be configured before the **lacp fast-switchover dampening** command is configured.

To configure LACP 1:1 redundancy, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *group_number* **Example:** `Device(config)# interface port-channel 40`	Selects an LACP port channel interface and enters interface configuration mode.
Step 4	**lacp fast-switchover** **Example:** `Device(config-if)# lacp fast-switchover`	Enables the LACP 1:1 Redundancy feature on the EtherChannel.

	Command or Action	Purpose
Step 5	**lacp max-bundle 1** **Example:** `Device(config-if)# lacp max-bundle 1`	Sets the maximum number of active member ports to be one. The only value that is supported with LACP 1:1 redundancy is 1.
Step 6	**lacp fast-switchover dampening** *seconds* **Example:** `Device(config-if)# lacp fast-switchover dampening 60`	(Optional) Enables the LACP 1:1 Hot Standby Dampening feature for this EtherChannel. The range for the time parameter is from 30 to 180 seconds.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Link Aggregation Control Protocol 1:1 Redundancy Fast Rate Timer

You can change the LACP timer rate to modify the duration of the LACP timeout. Use the **lacp rate** command to set the rate at which LACP control packets are received by an LACP-supported interface. You can change the timeout rate from the default rate (30 seconds) to the fast rate (1 second). This command is supported only on LACP-enabled interfaces.

To configure LACP 1:1 redundancy fast rate timer, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet***slot/port* **Example:** `Device(config)# interface gigabitEthernet 1/1`	Configures an interface and enters interface configuration mode.
Step 4	**lacp rate** {**normal** | **fast**} **Example:** `Device(config-if)# lacp rate fast`	Configures the rate at which LACP control packets are received by an LACP-supported interface. To reset the timeout rate to its default, use the **no lacp rate** command.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show lacp internal** **Example:** `Device# show lacp internal` `Device# show lacp counters`	Verifies your configuration.

Configuring Auto-LAG Globally

To configure Auto-LAG globally, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	[**no**] **port-channel auto** **Example:** `Device(config)# port-channel auto`	Enables the auto-LAG feature on a switch globally. Use the no form of this command to disable the auto-LAG feature on the switch globally. **Note** By default, the auto-LAG feature is enabled on the port.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show etherchannel auto** **Example:** `Device# show etherchannel auto`	Displays that EtherChannel is created automatically.

Configuring Auto-LAG on a Port Interface

To configure Auto-LAG on a port interface, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port interface to be enabled for auto-LAG, and enters interface configuration mode.
Step 4	[**no**] **channel-group auto** **Example:** `Device(config-if)# channel-group auto`	(Optional) Enables auto-LAG feature on individual port interface. Use the no form of this command to disable the auto-LAG feature on individual port interface. **Note** By default, the auto-LAG feature is enabled on the port.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show etherchannel auto** **Example:** `Device# show etherchannel auto`	Displays that EtherChannel is created automatically.

Configuring Persistence with Auto-LAG

You use the persistence command to convert the auto created EtherChannel into a manual one and allow you to add configuration on the existing EtherChannel.

To configure persistence with Auto-LAG, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**port-channel** *channel-number* **persistent** **Example:** `Device# port-channel 1 persistent`	Converts the auto created EtherChannel into a manual one and allows you to add configuration on the EtherChannel.
Step 3	**show etherchannel summary** **Example:** `Device# show etherchannel summary`	Displays the EtherChannel information.

Monitoring EtherChannel, Port Aggregation Protocol, and Link Aggregation Control Protocol Status

You can display EtherChannel, PAgP, and LACP status using the commands listed in this table.

Table 26: Commands for Monitoring EtherChannel, PAgP, and LACP Status

Command	Description
clear lacp { *channel-group-number* **counters** | **counters** }	Clears LACP channel-group information and traffic counters.
clear pagp { *channel-group-number* **counters** | **counters** }	Clears PAgP channel-group information and traffic counters.
show etherchannel [*channel-group-number* { **detail** | **load-balance** | **port** | **port-channel** | **protocol** | **summary** }] [**detail** | **load-balance** | **port** | **port-channel** | **protocol** | **auto** | **summary**]	Displays EtherChannel information in a brief, detailed, and one-line summary form. Also displays the load-balance or frame-distribution scheme, port, port-channel, protocol, and Auto-LAG information.
show pagp [*channel-group-number*] { **counters** | **internal** | **neighbor** }	Displays PAgP information such as traffic information, the internal PAgP configuration, and neighbor information.
show pagp [*channel-group-number*] **dual-active**	Displays the dual-active detection status.
show lacp [*channel-group-number*] { **counters** | **internal** | **neighbor** | **sys-id**}	Displays LACP information such as traffic information, the internal LACP configuration, and neighbor information.

Command	Description
show running-config	Verifies your configuration entries.
show etherchannel load-balance	Displays the load balance or frame distribution scheme among ports in the port channel.

Configuration Examples for EtherChannels

The following sections provide various configuration examples for EtherChannels:

Example: Configuring Layer 2 EtherChannels

This example shows how to configure an EtherChannel on a single switch . It assigns two ports as static-access ports in VLAN 10 to channel 5 with the PAgP mode **desirable**:

```
Device# configure terminal
Device(config)# interface range gigabitethernet2/0/1 -2
Device(config-if-range)# switchport mode access
Device(config-if-range)# switchport access vlan 10
Device(config-if-range)# channel-group 5 mode desirable non-silent
Device(config-if-range)# end
```

This example shows how to configure an EtherChannel on a single switch . It assigns two ports as static-access ports in VLAN 10 to channel 5 with the LACP mode **active :**

```
Device# configure terminal
Device(config)# interface range gigabitethernet2/0/1 -2
Device(config-if-range)# switchport mode access
Device(config-if-range)# switchport access vlan 10
Device(config-if-range)# channel-group 5 mode active
Device(config-if-range)# end
```

PoE or LACP negotiation errors may occur if you configure two ports from switch to the access point (AP). This scenario can be avoided if the port channel configuration is on the switch side. For more details, see the following example:

```
Device(config)# interface Port-channel1
Device(config-if)# switchport access vlan 20
Device(config-if)# switchport mode access
Device(config-if)# switchport nonegotiate
Device(config-if)# no port-channel standalone-disable
Device(config-if)# spanning-tree portfast
```

Note If the port reports LACP errors on port flap, you should include the following command as well: **no errdisable detect cause pagp-flap**

Example: Configuring Layer 3 EtherChannels

This example shows how to configure a Layer 3 EtherChannel. It assigns two ports to channel 5 with the LACP mode **active**:

```
Device# configure terminal
Device(config)# interface range gigabitethernet2/0/1 -2
Device(config-if-range)# no ip address
Device(config-if-range)# no switchport
Device(config-if-range)# channel-group 5 mode active
Device(config-if-range)# end
```

Example: Configuring Link Aggregation Control Protocol Hot-Standby Ports

This example shows how to configure an EtherChannel (port channel 2) that will be active when there are at least three active ports, will comprise up to seven active ports and the remaining ports (up to nine) as hot-standby ports:

```
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# port-channel min-links 3
Device(config-if)# lacp max-bundle 7
```

Example: Configuring Link Aggregation Control Protocol 1:1 Redundancy

This example shows how to configure the LACP 1:1 Redundancy feature on the EtherChannel:

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 40
Device(config-if)# lacp fast-switchover
Device(config-if)# lacp max-bundle 1
Device(config-if)# lacp fast-switchover dampening 60
Device(config-if)# end
```

This is a sample output from the **show lacp internal** command:

```
Device# show lacp 1 internal

Flags: S - Device is requesting Slow LACPDUs
       F - Device is requesting Fast LACPDUs
       A - Device is in Active mode
       P - Device is in Passive mode

Channel group 1,[146 s left to exit dampening state]
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
GE1/1     FA      hot-sby   30000*        0x1       0x1     0x103       0x7
GE1/2     SA      bndl      32768         0x1       0x1     0x102       0x3D
```

Example: Configuring Standalone Mode on EtherChannel

This example shows how to configure the Standalone mode or Independent mode on an Port channel:

```
Device(config)# interface port-channel 1
Device(config-if)# no port-channel standalone-disable
Device(config-if)# end
```

This example shows how to verify the configuration of the Standalone mode on a Port Channel interface:

```
Device# show running-config interface port-channel 1
Building configuration...
Current configuration:
```

```
!
interface Port-channel1
  no ip address
  no switchport
  no port-channel standalone-disable
end
```

Example: Configuring Auto LAG

This example shows how to configure Auto-LAG on a switch

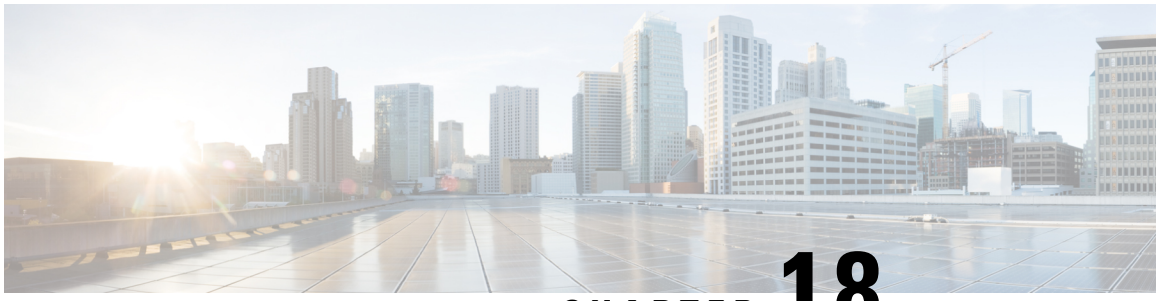
```
Device> enable
Device# configure terminal
Device(config)# port-channel auto
Device(config-if)# end
Device#  show etherchannel auto
```

This example shows the summary of EtherChannel that was created automatically.

```
Device# show etherchannel auto
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port
        A - formed by Auto LAG

Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SUA)        LACP      Gi1/1(P) Gi1/2(P) Gi1/3(P)
```

This example shows the summary of auto EtherChannel after executing the **port-channel** 1 **persistent** command.

```
Device# port-channel 1 persistent

Device# show etherchannel summary
Switch# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port
        A - formed by Auto LAG

Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SU)         LACP      Gi1/1(P) Gi1/2(P) Gi1/3(P)
```



CHAPTER 18

Configuring UniDirectional Link Detection

Restrictions for Configuring UniDirectional Link Detection

The following are restrictions for configuring UniDirectional Link Detection (UDLD):

- A UDLD-capable port can't detect a unidirectional link if it's connected to a UDLD-incapable port of another device.
- When configuring the mode (normal or aggressive), make sure that the same mode is configured on both sides of the link.
- Alert option for UDLD ports are not supported.

Caution Loop guard works only on point-to-point links. We recommend that each end of the link has a directly connected device that is running STP.

Information About UniDirectional Link Detection

UniDirectional Link Detection (UDLD) is a Layer 2 protocol that enables devices that are connected through fiber-optic or twisted-pair Ethernet cables to monitor the physical configuration of the cables and detect when a unidirectional link exists. All connected devices must support UDLD for the protocol to successfully identify and disable unidirectional links. When UDLD detects a unidirectional link, it disables the affected port and alerts you. Unidirectional links can cause a variety of problems, including spanning-tree topology loops.

Modes of Operation

UDLD supports two modes of operation: normal (the default) and aggressive. In normal mode, UDLD can detect unidirectional links due to misconnected ports on fiber-optic connections. In aggressive mode, UDLD

can also detect unidirectional links due to one-way traffic on fiber-optic and twisted-pair links and to misconnected ports on fiber-optic links.

In normal and aggressive modes, UDLD works with the Layer 1 mechanisms to learn the physical status of a link. At Layer 1, autonegotiation takes care of physical signaling and fault detection. UDLD performs tasks that autonegotiation cannot perform, such as detecting the identities of neighbors and shutting down misconnected ports. When you enable both autonegotiation and UDLD, the Layer 1 and Layer 2 detections work together to prevent physical and logical unidirectional connections and the malfunctioning of other protocols.

A unidirectional link occurs whenever traffic sent by a local device is received by its neighbor but traffic from the neighbor is not received by the local device.

Normal Mode

In normal mode, UDLD detects a unidirectional link when fiber strands in a fiber-optic port are misconnected and the Layer 1 mechanisms do not detect this misconnection. If the ports are connected correctly but the traffic is one way, UDLD does not detect the unidirectional link because the Layer 1 mechanism, which is supposed to detect this condition, does not do so. In this case, the logical link is considered undetermined, and UDLD does not disable the port.

When UDLD is in normal mode, if one of the fiber strands in a pair is disconnected, as long as autonegotiation is active, the link does not stay up because the Layer 1 mechanisms detects a physical problem with the link. In this case, UDLD does not take any action and the logical link is considered undetermined.

Aggressive Mode

In aggressive mode, UDLD detects a unidirectional link by using the previous detection methods. UDLD in aggressive mode can also detect a unidirectional link on a point-to-point link on which no failure between the two devices is allowed. It can also detect a unidirectional link when one of these problems exists:

- On fiber-optic or twisted-pair links, one of the ports cannot send or receive traffic.
- On fiber-optic or twisted-pair links, one of the ports is down while the other is up.
- One of the fiber strands in the cable is disconnected.

In these cases, UDLD disables the affected port.

In a point-to-point link, UDLD hello packets can be considered as a heart beat whose presence guarantees the health of the link. Conversely, the loss of the heart beat means that the link must be shut down if it is not possible to reestablish a bidirectional link.

If both fiber strands in a cable are working normally from a Layer 1 perspective, UDLD in aggressive mode detects whether those fiber strands are connected correctly and whether traffic is flowing bidirectionally between the correct neighbors. This check cannot be performed by autonegotiation because autonegotiation operates at Layer 1.

Methods to Detect Unidirectional Links

UDLD operates by using two methods:

- Neighbor database maintenance
- Event-driven detection and echoing

Neighbor Database Maintenance

UDLD learns about other UDLD-capable neighbors by periodically sending a hello packet (also called an advertisement or probe) on every active port to keep each device informed about its neighbors.

When the device receives a hello message, it caches the information until the age time (hold time or time-to-live) expires. If the device receives a new hello message before an older cache entry ages, the device replaces the older entry with the new one.

Whenever a port is disabled and UDLD is running, whenever UDLD is disabled on a port, or whenever the device is reset, UDLD clears all existing cache entries for the ports that are affected by the configuration change. UDLD sends at least one message to inform the neighbors to flush the part of their caches affected by the status change. The message is intended to keep the caches synchronized.

Note An interface does not support multiple UDLD neighbors. If an ingress UDLD protocol data unit (PDU) has multiple device IDs in echo type, length and value (TLV), the interface enters the error-disabled state.

Event-Driven Detection and Echoing

UDLD relies on echoing as its detection operation. Whenever a UDLD device learns about a new neighbor or receives a resynchronization request from an out-of-sync neighbor, it restarts the detection window on its side of the connection and sends echo messages in reply. Because this behavior is the same on all UDLD neighbors, the sender of the echoes expects to receive an echo in reply.

If the detection window ends and no valid reply message are received, the link might shut down, depending on the UDLD mode. When UDLD is in normal mode, the link might be considered undetermined and might not be shut down. When UDLD is in aggressive mode, the link is considered unidirectional, and the port is disabled.

UniDirectional Link Detection Reset Options

If an interface becomes disabled by UDLD, you can use one of the following options to reset UDLD:

- The **udld reset** interface configuration command.
- The **shutdown** interface configuration command followed by the **no shutdown** interface configuration command restarts the disabled port.
- The **no udld** {**aggressive** | **enable**} global configuration command followed by the **udld** {**aggressive** | **enable**} global configuration command reenables the disabled ports.
- The **no udld port** interface configuration command followed by the **udld port** [**aggressive**] interface configuration command reenables the disabled fiber-optic port.
- The **errdisable recovery cause udld** global configuration command enables the timer to automatically recover from the UDLD error-disabled state, and the **errdisable recovery interval** *interval* global configuration command specifies the time to recover from the UDLD error-disabled state.

The **udld port disable** command disables UDLD on fiber-optic LAN ports.

Note This command is only supported on fiber-optic LAN ports.

Default UniDirectional Link Detection Configuration

Table 27: Default UDLD Configuration

Feature	Default Setting
UDLD global enable state	Globally disabled
UDLD per-port enable state for fiber-optic media	Disabled on all Ethernet fiber-optic ports
UDLD per-port enable state for twisted-pair (copper) media	Disabled on all Ethernet 10/100 and 1000BASE-TX ports
UDLD aggressive mode	Disabled

How to Configure UniDirectional Link Detection

The following sections provide information about configuring UDLD:

Enabling UniDirectional Link Detection Globally

Follow these steps to enable UDLD in the aggressive or normal mode and to set the configurable message timer on all fiber-optic ports on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**udld** {**aggressive** | **enable** | **message time** *message-timer-interval*} **Example:** `Device(config)# udld enable message time 10`	Specifies the UDLD mode of operation: • **aggressive**—Enables UDLD in aggressive mode on all fiber-optic ports. • **enable**—Enables UDLD in normal mode on all fiber-optic ports on the device. UDLD is disabled by default. An individual interface configuration overrides the setting of the **udld enable** global configuration command.

	Command or Action	Purpose
		• **message time** *message-timer-interval*—Configures the period of time between UDLD probe messages on ports that are in the advertisement phase and are detected to be bidirectional. The range is from 1 to 90 seconds; the default value is 15. **Note** This command affects fiber-optic ports only. Use the **udld** interface configuration command to enable UDLD on other port types. Use the **no** form of this command, to disable UDLD.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Enabling UniDirectional Link Detection on an Interface

Follow these steps either to enable UDLD in the aggressive or normal mode or to disable UDLD on a port.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be enabled for UDLD, and enters interface configuration mode.
Step 4	**udld port** [**aggressive**] **Example:** `Device(config-if)# udld port aggressive`	UDLD is disabled by default. • **udld port**—Enables UDLD in normal mode on the specified port.

	Command or Action	Purpose
		• **udld port aggressive**—(Optional) Enables UDLD in aggressive mode on the specified port. **Note** Use the **no udld port** interface configuration command to disable UDLD on a specified fiber-optic port.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Disabling UniDirectional Link Detection on Fiber-Optic LAN Interfaces

To disable UDLD on Fiber-optic LAN interfaces, perform this procedure:

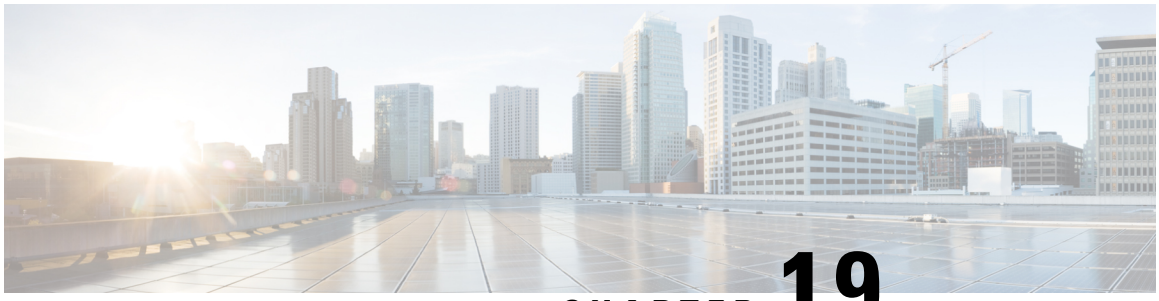
Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Configures an interface and enters interface configuration mode.
Step 4	**udld port disable** **Example:** `Device(config-if)# udld port disable`	Disables UDLD on a fiber-optic LAN port. • The **udld port disable** command is only supported on fiber-optic LAN ports. • The **no udld port disable** command reverts to the **udld enable** global configuration command setting.
Step 5	**end** **Example:**	Exits interface configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	

Monitoring and Maintaining UniDirectional Link Detection

Command	Purpose
show udld [*interface-id* | **neighbors**]	Displays the UDLD status for the specified port or for all ports.



CHAPTER 19

Configuring Layer 2 Protocol Tunneling

Prerequisites for Layer 2 Protocol Tunneling

The following sections list prerequisites and considerations for configuring Layer 2 protocol tunneling.

To configure Layer 2 point-to-point tunneling to facilitate the automatic creation of EtherChannels, you need to configure both the SP (service-provider) edge switch and the customer device.

Information About Layer 2 Protocol Tunneling

The following sections provide information about Layer 2 protocol tunneling:

Layer 2 Protocol Tunneling Overview

Customers at different sites that are connected across a service-provider network need to use various Layer 2 protocols to scale their topologies to include all remote sites, as well as the local sites. STP must run properly, and every VLAN should build a proper spanning tree that includes the local site and all remote sites across the service-provider network. Cisco Discovery Protocol (CDP) must discover neighboring Cisco devices from local and remote sites. VLAN Trunking Protocol (VTP) must provide consistent VLAN configuration throughout all sites in the customer network.

When protocol tunneling is enabled, edge device on the inbound side of the service-provider network encapsulate Layer 2 protocol packets with a special MAC address and send them across the service-provider network. Core devices in the network do not process these packets but forward them as normal packets. Layer 2 protocol data units (PDUs) for CDP, STP, or VTP cross the service-provider network and are delivered to customer devices on the outbound side of the service-provider network. Identical packets are received by all customer ports on the same VLANs with these results:

- Users on each of a customer's sites can properly run STP, and every VLAN can build a correct spanning tree based on parameters from all sites and not just from the local site.
- CDP discovers and shows information about the other Cisco devices that are connected through the service-provider network.
- VTP provides consistent VLAN configuration throughout the customer network, propagating to all devices through the service provider.

Layer 2 protocol tunneling can be used independently or can enhance IEEE 802.1Q tunneling. If protocol tunneling is not enabled on IEEE 802.1Q tunneling ports, remote devices at the receiving end of the service-provider network do not receive the PDUs and cannot properly run STP, CDP, and VTP. When protocol tunneling is enabled, Layer 2 protocols within each customer's network are totally separate from those running within the service-provider network. Customer devices on different sites that send traffic through the service-provider network with IEEE 802.1Q tunneling achieve complete knowledge of the customer's VLAN. If IEEE 802.1Q tunneling is not used, you can still enable Layer 2 protocol tunneling by connecting to the customer device through access ports and by enabling tunneling on the service-provider access port.

For example, in the following figure (Layer 2 Protocol Tunneling), Customer X has four switches in the same VLAN, that are connected through the service-provider network. If the network does not tunnel PDUs, switches on the far ends of the network cannot properly run STP, CDP, and VTP. For example, STP for a VLAN on a switch in Customer X, Site 1, will build a spanning tree on the switches at that site without considering convergence parameters based on Customer X's switch in Site 2. This could result in the topology that is shown in the Layer 2 Network Topology without Proper Convergence figure.

Figure 23: Layer 2 Protocol Tunneling

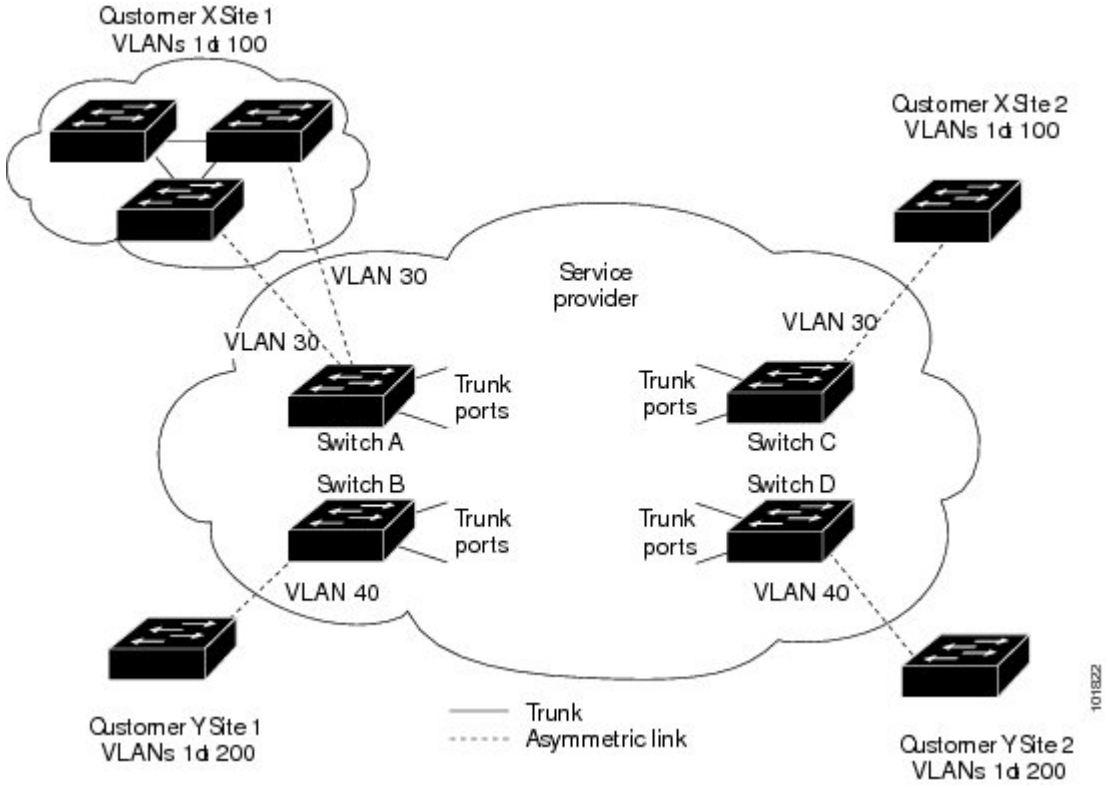
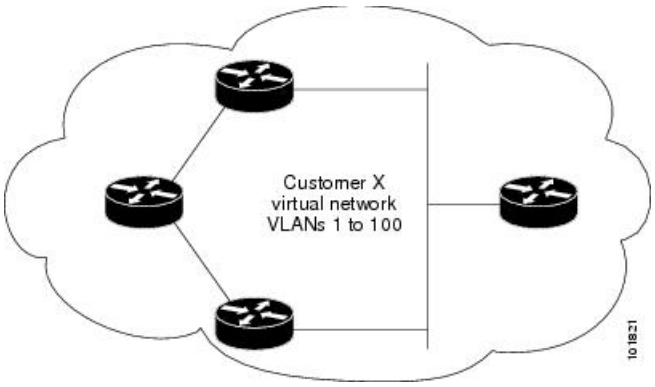


Figure 24: Layer 2 Network Topology Without Proper Convergence



Layer 2 Protocol Tunneling on Ports

You can enable Layer 2 protocol tunneling (by protocol) on the ports that are connected to the customer in the edge devices of the service-provider network. The service-provider edge devices connected to the customer device perform the tunneling process. Edge device tunnel ports are connected to customer IEEE 802.1Q trunk ports. Edge device access ports are connected to customer access ports. The edge devices connected to the customer device perform the tunneling process.

You can enable Layer 2 protocol tunneling on ports that are configured as access ports or tunnel ports or trunk ports. You cannot enable Layer 2 protocol tunneling on ports that are configured in either **switchport mode dynamic auto** mode (the default mode) or **switchport mode dynamic desirable** mode.

The device supports Layer 2 protocol tunneling for CDP, STP, and VTP. For emulated point-to-point network topologies, it also supports PAgP, LACP, LLDP, and UDLD protocols.

Note PAgP, LACP, and UDLD protocol tunneling are only intended to emulate a point-to-point topology. An erroneous configuration that sends tunneled packets to many ports could lead to a network failure.

When the Layer 2 PDUs that entered the service-provider inbound edge device through a Layer 2 protocol-enabled port exit through the trunk port into the service-provider network, the device overwrites the customer PDU-destination MAC address with a well-known Cisco proprietary multicast address (01-00-0c-cd-cd-d0). If IEEE 802.1Q tunneling is enabled, packets are also double-tagged; the outer tag is the customer metro tag, and the inner tag is the customer's VLAN tag. The core devices ignore the inner tags and forward the packet to all trunk ports in the same metro VLAN. The edge devices on the outbound side restore the proper Layer 2 protocol and MAC address information and forward the packets to all tunnel or access ports in the same metro VLAN. Therefore, the Layer 2 PDUs remain intact and are delivered across the service-provider infrastructure to the other side of the customer network.

See the Layer 2 Protocol Tunneling figure in Layer 2 Protocol Tunneling Overview, with Customer X and Customer Y in access VLANs 30 and 40, respectively. Asymmetric links connect the customers in Site 1 to edge switches in the service-provider network. The Layer 2 PDUs (for example, BPDUs) coming into Switch B from Customer Y in Site 1 are forwarded to the infrastructure as double-tagged packets with the well-known MAC address as the destination MAC address. These double-tagged packets have the metro VLAN tag of 40, as well as an inner VLAN tag (for example, VLAN 100). When the double-tagged packets enter Switch D, the outer VLAN tag 40 is removed, the well-known MAC address is replaced with the respective Layer 2

protocol MAC address, and the packet is sent to Customer Y on Site 2 as a single-tagged frame in VLAN 100.

You can also enable Layer 2 protocol tunneling on access ports on the edge switch that is connected to access or trunk ports on the customer switch. In this case, the encapsulation and decapsulation process are the same as described in the previous paragraph, except that the packets are not double-tagged in the service-provider network. The single tag is the customer-specific access VLAN tag.

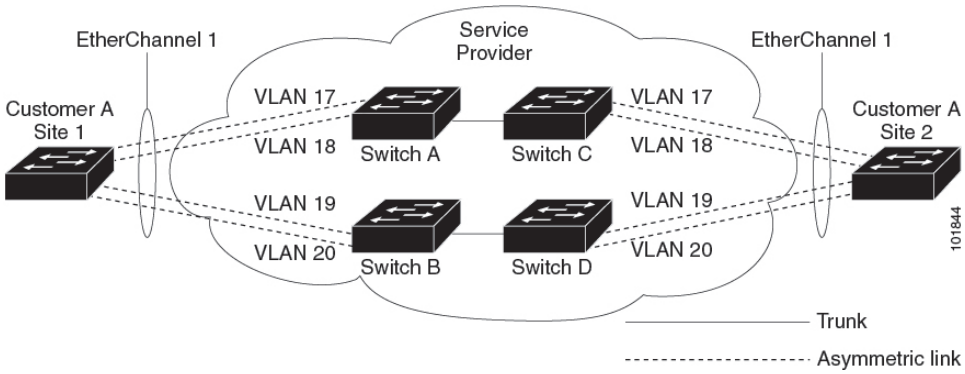
Layer 2 Protocol Tunneling for EtherChannels

In an SP network, you can use Layer 2 protocol tunneling to enhance the creation of EtherChannels by emulating a point-to-point network topology. When you enable protocol tunneling (PAgP or LACP) on the SP switch, remote customer switches receive the PDUs and can negotiate the automatic creation of EtherChannels.

For example, in the following figure (Layer 2 Protocol Tunneling for EtherChannels), Customer A has two switches in the same VLAN that are connected through the SP network. When the network tunnels PDUs, switches on the far ends of the network can negotiate the automatic creation of EtherChannels without needing dedicated lines.

While configuring Layer 2 Protocol Tunneling on trunk ports, both the trunk ports on the SP edge device should be configured with different native VLANs. The native VLAN of one trunk port should not be in the list of allowed VLANs of the other trunk port to avoid loops.

Figure 25: Layer 2 Protocol Tunneling for EtherChannels



Default Layer 2 Protocol Tunneling Configuration

The following table shows the default Layer 2 protocol tunneling configuration.

Table 28: Default Layer 2 Ethernet Interface VLAN Configuration

Feature	Default Setting
Layer 2 protocol tunneling	Disabled.
Shutdown threshold	None set.
Drop threshold	None set.

How to Configure Layer 2 Protocol Tunneling

The following section provides configuration information on how to configure a layer 2 protocol tunnel:

Configuring Layer 2 Protocol Tunneling

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface that is connected to the phone, and enters interface configuration mode.
Step 4	Use one of the following: • **switchport mode dot1q-tunnel** • **switchport mode trunk** **Example:** `Device(config-if)# switchport mode dot1q-tunnel` or `Device(config-if)# switchport mode trunk`	Configures the interface as an IEEE 802.1Q tunnel port or a trunk port.
Step 5	**l2protocol-tunnel** [**lldp** | **point-to-point** | **stp** | **vtp**] **Example:** `Device(config-if)# l2protocol-tunnel lldp`	Enables protocol tunneling for the desired protocol. If no keyword is entered, tunneling is enabled for all four Layer 2 protocols. **Note** Use the **no l2protocol-tunnel** [**lldp** | **point-to-point** | **stp** | **vtp**] interface configuration command to disable protocol tunneling for one of the Layer 2 protocols or for all three.
Step 6	**l2protocol-tunnel shutdown-threshold** [*packet_second_rate_value* | **lldp** **point-to-point** | **stp** | **vtp**]	(Optional) Configures the threshold for packets-per-second that are accepted for encapsulation. The interface is disabled if the

	Command or Action	Purpose
	Example: `Device(config-if)# l2protocol-tunnel shutdown-threshold 100`	configured threshold is exceeded. If no protocol option is specified, the threshold applies to each of the tunneled Layer 2 protocol types. The range is 1 to 4096. The default is to have no threshold configured. **Note** If you also set a drop threshold on this interface, the **shutdown-threshold** value must be greater than or equal to the **drop-threshold** value. **Note** Use the **no l2protocol-tunnel shutdown-threshold** [*packet_second_rate_value* | **lldp**| **point-to-point** | **stp** | **vtp**] and the **no l2protocol-tunnel drop-threshold** [*packet_second_rate_value* | **lldp**| **point-to-point** |**stp** | **vtp**] commands to return the shutdown and drop thresholds to the default settings.
Step 7	**l2protocol-tunnel drop-threshold** [*packet_second_rate_value* | **lldp** | **point-to-point** | **stp** | **vtp**] **Example:** `Device(config-if)# l2protocol-tunnel drop-threshold 100 lldp`	(Optional) Configures the threshold for packets-per-second that are accepted for encapsulation. The interface drops packets if the configured threshold is exceeded. If no protocol option is specified, the threshold applies to each of the tunneled Layer 2 protocol types. The range is 1 to 4096. The default is to have no threshold configured. **Note** If you also set a shutdown threshold on this interface, the **drop-threshold** value must be less than or equal to the **shutdown-threshold** value. **Note** Use the **no l2protocol-tunnel shutdown-threshold** [**lldp**| **point-to-point** | **stp** | **vtp**] and the **no l2protocol-tunnel drop-threshold** [**stp** | **vtp**] commands to return the shutdown and drop thresholds to the default settings.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.

	Command or Action	Purpose
Step 9	**errdisable recovery cause l2ptguard** **Example:** `Device(config)# errdisable recovery cause l2ptguard`	(Optional) Configures the recovery mechanism from a Layer 2 maximum-rate error so that the interface is reenabled and can try again. Errdisable recovery is disabled by default; when enabled, the default time interval is 300 seconds.
Step 10	**spanning-tree bpdufilter enable** **Example:** `Device(config)# spanning-tree bpdufilter enable`	Inserts a BPDU filter for spanning tree. **Note** While configuring Layer 2 Protocol Tunneling on a trunk port, you must enable a BPDU filter for spanning tree.
Step 11	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 12	**show l2protocol** **Example:** `Device# show l2protocol`	Displays the Layer 2 tunnel ports on the device, including the protocols configured, the thresholds, and the counters.
Step 13	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

How to Configure Layer 2 Protocol Tunneling for EtherChannels

For EtherChannels, you need to configure both the SP (service-provider) edge devices and the customer devices for Layer 2 protocol tunneling. The following sections provide configuration information on how to configure the SP edge device and how to configure the customer device:

Configuring the SP Edge Switch

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface that is connected to the phone, and enters interface configuration mode.
Step 4	**switchport trunk native vlan** *vlan-id* **Example:** `Device(config-if)# switchport trunk native vlan 2`	Configures the native VLAN. **Note** While configuring Layer 2 Protocol Tunneling for EtherChannels on trunk ports, you must configure different native VLANs on both trunk ports on the SP edge device.
Step 5	**switchport trunk allowed vlan** *vlan-id list* **Example:** `Device(config-if)# switchport trunk allowed vlan 1,2,4-3003,3005-4094`	Specifies the list of allowed VLANs. **Note** While configuring Layer 2 Protocol Tunneling for EtherChannels on trunk ports, you must ensure that the native VLAN of one trunk port of the SP edge device should not be in the list of allowed VLANs of the other trunk port to avoid loops.
Step 6	Use one of the following: • **switchport mode dot1q-tunnel** • **switchport mode trunk** **Example:** `Device(config-if)# switchport mode dot1q-tunnel` or `Device(config-if)# switchport mode trunk`	Configures the interface as an IEEE 802.1Q tunnel port or as a trunk port.
Step 7	**l2protocol-tunnel point-to-point** [**pagp** | **lacp** | **udld**] **Example:** `Device(config-if)# l2protocol-tunnel point-to-point pagp`	(Optional) Enables point-to-point protocol tunneling for the desired protocol. If no keyword is entered, tunneling is enabled for all three protocols. **Note** To avoid a network failure, make sure that the network is a point-to-point topology before you enable tunneling for PAgP, LACP, or UDLD packets. **Note**

	Command or Action	Purpose
		Use the **no l2protocol-tunnel** [**point-to-point** [**pagp** | **lacp** | **udld**]] interface configuration command to disable point-to-point protocol tunneling for one of the Layer 2 protocols or for all three.
Step 8	**l2protocol-tunnel shutdown-threshold** [**point-to-point** [**pagp** | **lacp** | **udld**]] *value* **Example:** `Device(config-if)# l2protocol-tunnel shutdown-threshold point-to-point pagp 100`	(Optional) Configures the threshold for packets-per-second that are accepted for encapsulation. The interface is disabled if the configured threshold is exceeded. If no protocol option is specified, the threshold applies to each of the tunneled Layer 2 protocol types. The range is 1 to 4096. The default is to have no threshold configured. **Note** If you also set a drop threshold on this interface, the **shutdown-threshold** value must be greater than or equal to the **drop-threshold** value. **Note** Use the **no l2protocol-tunnel shutdown-threshold** [**point-to-point** [**pagp** | **lacp** | **udld**]] and the **no l2protocol-tunnel drop-threshold** [[**point-to-point** [**pagp** | lacp | **udld**]] commands to return the shutdown and drop thresholds to the default settings.
Step 9	**l2protocol-tunnel drop-threshold** [**point-to-point** [**pagp** | **lacp** | **udld**]] *value* **Example:** `Device(config-if)# l2protocol-tunnel drop-threshold point-to-point pagp 500`	(Optional) Configures the threshold for packets-per-second that are accepted for encapsulation. The interface drops packets if the configured threshold is exceeded. If no protocol option is specified, the threshold applies to each of the tunneled Layer 2 protocol types. The range is 1 to 4096. The default is to have no threshold configured. **Note** If you also set a shutdown threshold on this interface, the **drop-threshold** value must be less than or equal to the **shutdown-threshold** value.
Step 10	**no cdp enable** **Example:** `Device(config-if)# no cdp enable`	Disables CDP on the interface.

	Command or Action	Purpose
Step 11	**spanning-tree bpdu filter enable** **Example:** `Device(config-if)# spanning-tree bpdu filter enable`	Enables BPDU filtering on the interface.
Step 12	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 13	**errdisable recovery cause l2ptguard** **Example:** `Device(config)# errdisable recovery cause l2ptguard`	(Optional) Configures the recovery mechanism from a Layer 2 maximum-rate error so that the interface is reenabled and can try again. Errdisable recovery is disabled by default; when enabled, the default time interval is 300 seconds.
Step 14	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 15	**show l2protocol** **Example:** `Device# show l2protocol`	Displays the Layer 2 tunnel ports on the device, including the protocols configured, the thresholds, and the counters.
Step 16	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring the Customer Device

Before you begin

For EtherChannels, you need to configure both the SP edge device and the customer device for Layer 2 protocol tunneling.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface that is connected to the phone, and enters interface configuration mode.
Step 4	**switchport trunk encapsulation dot1q** **Example:** `Device(config-if)# switchport trunk encapsulation dot1q`	Sets the trunking encapsulation format to IEEE 802.1Q.
Step 5	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Enables trunking on the interface.
Step 6	**udld port** **Example:** `Device(config-if)# udld port`	Enables UDLD in normal mode on the interface.
Step 7	**channel-group** *channel-group-number* **mode desirable** **Example:** `Device(config-if)# channel-group 25 mode desirable`	Assigns the interface to a channel group, and specifies desirable for the PAgP mode.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 9	**interface port-channel port-channel** *number* **Example:** `Device(config)# interface port-channel port-channel 25`	Enters port-channel interface mode.
Step 10	**shutdown** **Example:** `Device(config)# shutdown`	Shuts down the interface.
Step 11	**no shutdown** **Example:** `Device(config)# no shutdown`	Enables the interface.

	Command or Action	Purpose
Step 12	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 13	**show l2protocol** **Example:** `Device# show l2protocol`	Displays the Layer 2 tunnel ports on the device, including the protocols configured, the thresholds, and the counters.
Step 14	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file. **Note** Use the **no switchport mode trunk**, the **no udld enable**, and the **no channel group** *channel-group-number* **mode desirable** interface configuration commands to return the interface to the default settings.

Configuration Examples for Layer 2 Protocol Tunneling

The following sections provide various configuration examples for layer 2 protocol tunneling:

Example: Configuring Layer 2 Protocol Tunneling

The following example shows how to configure Layer 2 protocol tunneling for Cisco Discovery Protocol, STP, and VTP and to verify the configuration.

```
Device(config)# interface gigabitethernet1/1
Device(config-if)# l2protocol-tunnel cdp
Device(config-if)# l2protocol-tunnel stp
Device(config-if)# l2protocol-tunnel vtp
Device(config-if)# l2protocol-tunnel shutdown-threshold 1500
Device(config-if)# l2protocol-tunnel drop-threshold 1000
Device(config-if)# exit

Device(config)# end
Device# show l2protocol

Port Protocol Shutdown Drop Encapsulation Decapsulation Drop
Threshold Threshold Counter Counter Counter
------- -------- --------- --------- ------------- ------------- -------------
Gi1/1 cdp 1500 1000 2288 2282 0
stp 1500 1000 116 13 0
vtp 1500 1000 3 67 0
pagp ---- ---- 0 0 0
lacp ---- ---- 0 0 0
udld ---- ---- 0 0 0
```

Examples: Configuring the SP Edge and Customer Switches

This example shows how to configure the SP edge switch 1 and edge switch 2.

SP edge switch 1 configuration:

```
Device(config)# interface gigabitethernet1/1
Device(config-if)# switchport access vlan 17
Device(config-if)# switchport mode dot1q-tunnel
Device(config-if)# l2protocol-tunnel point-to-point pagp
Device(config-if)# l2protocol-tunnel point-to-point udld
Device(config-if)# l2protocol-tunnel drop-threshold point-to-point pagp 1000
Device(config-if)# exit
Device(config)# interface gigabitethernet1/2
Device(config-if)# switchport access vlan 18
Device(config-if)# switchport mode dot1q-tunnel
Device(config-if)# l2protocol-tunnel point-to-point pagp
Device(config-if)# l2protocol-tunnel point-to-point udld
Device(config-if)# l2protocol-tunnel drop-threshold point-to-point pagp 1000
Device(config-if)# exit
Device(config)# interface gigabitethernet1/3
Device(config-if)# switchport trunk encapsulation isl
Device(config-if)# switchport mode trunk
```

SP edge switch 2 configuration:

```
Device(config)# interface gigabitethernet1/1
Device(config-if)# switchport access vlan 19
Device(config-if)# switchport mode dot1q-tunnel
Device(config-if)# l2protocol-tunnel point-to-point pagp
Device(config-if)# l2protocol-tunnel point-to-point udld
Device(config-if)# l2protocol-tunnel drop-threshold point-to-point pagp 1000
Device(config-if)# exit
Device(config)# interface gigabitethernet1/2
Device(config-if)# switchport access vlan 20
Device(config-if)# switchport mode dot1q-tunnel
Device(config-if)# l2protocol-tunnel point-to-point pagp
Device(config-if)# l2protocol-tunnel point-to-point udld
Device(config-if)# l2protocol-tunnel drop-threshold point-to-point pagp 1000
Device(config-if)# exit
Device(config)# interface gigabitethernet1/3
Device(config-if)# switchport trunk encapsulation isl
Device(config-if)# switchport mode trunk
```

This example shows how to configure the customer switch at Site 1. Interfaces 1, 2, 3, and 4 are set for IEEE 802.1Q trunking, UDLD is enabled, EtherChannel group 1 is enabled, and the port channel is shut down and then enabled to activate the EtherChannel configuration.

```
Device(config)# interface gigabitethernet1/1
Device(config-if)# switchport trunk encapsulation dot1q
Device(config-if)# switchport mode trunk
Device(config-if)# udld enable
Device(config-if)# channel-group 1 mode desirable
Device(config-if)# exit
Device(config)# interface gigabitethernet1/2
Device(config-if)# switchport trunk encapsulation dot1q
Device(config-if)# switchport mode trunk
Device(config-if)# udld enable
```

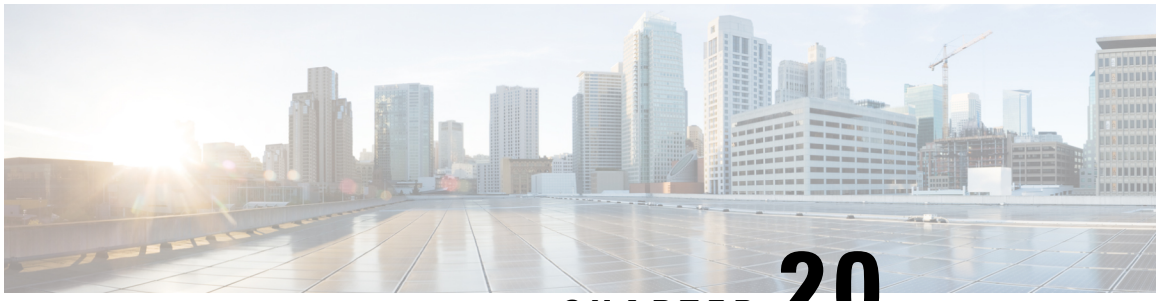
```
Device(config-if)# channel-group 1 mode desirable
Device(config-if)# exit
Device(config)# interface gigabitethernet1/3
Device(config-if)# switchport trunk encapsulation dot1q
Device(config-if)# switchport mode trunk
Device(config-if)# udld enable
Device(config-if)# channel-group 1 mode desirable
Device(config-if)# exit
Device(config)# interface gigabitethernet1/4
Device(config-if)# switchport trunk encapsulation dot1q
Device(config-if)# switchport mode trunk
Device(config-if)# udld enable
Device(config-if)# channel-group 1 mode desirable
Device(config-if)# exit
Device(config)# interface port-channel 1
Device(config-if)# shutdown
Device(config-if)# no shutdown
Device(config-if)# exit
```

Monitoring Tunneling Status

The following table describes the commands used to monitor tunneling status.

Table 29: Commands for Monitoring Tunneling

Command	Purpose
clear l2protocol-tunnel counters	Clears the protocol counters on Layer 2 protocol tunneling ports.
show dot1q-tunnel	Displays IEEE 802.1Q tunnel ports on the device.
show dot1q-tunnel interface *interface-id*	Verifies if a specific interface is a tunnel port.
show l2protocol-tunnel	Displays information about Layer 2 protocol tunneling ports.
show errdisable recovery	Verifies if the recovery timer from a Layer 2 protocol-tunnel error disable state is enabled.
show l2protocol-tunnel interface *interface-id*	Displays information about a specific Layer 2 protocol tunneling port.
show l2protocol-tunnel summary	Displays only Layer 2 protocol summary information.
show vlan dot1q tag native	Displays the status of native VLAN tagging on the device.



CHAPTER 20

Configuring IEEE 802.1Q Tunneling

Information About IEEE 802.1Q Tunneling

The IEEE 802.1Q Tunneling feature is designed for service providers who carry traffic of multiple customers across their networks and are required to maintain the VLAN and Layer 2 protocol configurations of each customer without impacting the traffic of other customers.

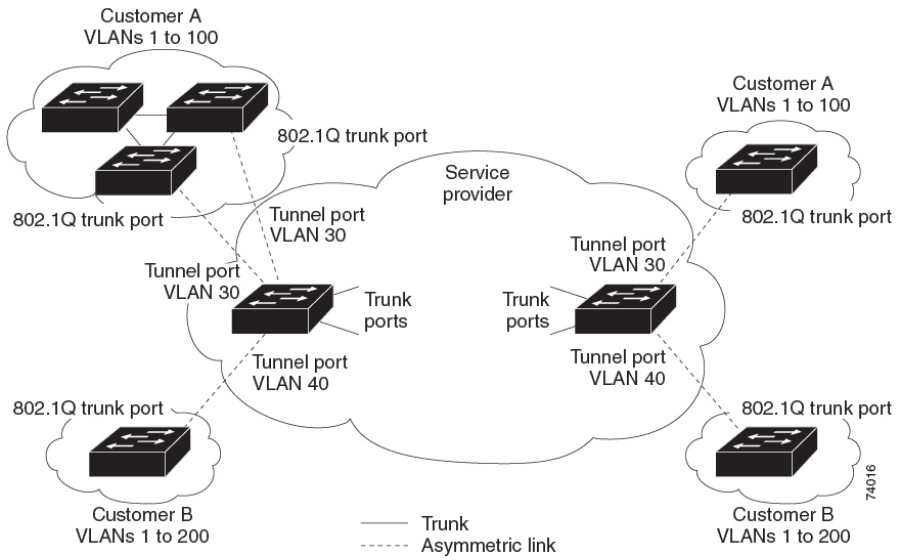
IEEE 802.1Q Tunnel Ports in a Service Provider Network

Business customers of service providers often have specific requirements for VLAN IDs and the number of VLANs to be supported. The VLAN ranges required by different customers in the same service-provider network might overlap, and traffic of customers through the infrastructure might be mixed. Assigning a unique range of VLAN IDs to each customer would restrict customer configurations and could easily exceed the VLAN limit (4096) of the IEEE 802.1Q specification.

Using the IEEE 802.1Q tunneling feature, service providers can use a single VLAN to support customers who have multiple VLANs. Customer VLAN IDs are preserved, and traffic from different customers is segregated within the service-provider network, even when they appear to be in the same VLAN. Using IEEE 802.1Q tunneling expands VLAN space by using a VLAN-in-VLAN hierarchy and retagging the tagged packets. A port configured to support IEEE 802.1Q tunneling is called a tunnel port. When you configure tunneling, you assign a tunnel port to a VLAN ID that is dedicated to tunneling. Each customer requires a separate service-provider VLAN ID, but that VLAN ID supports all of the customer's VLANs.

Customer traffic tagged in the normal way with appropriate VLAN IDs comes from an IEEE 802.1Q trunk port on the customer device and into a tunnel port on the service-provider edge device. The link between the customer device and the edge device is asymmetric because one end is configured as an IEEE 802.1Q trunk port, and the other end is configured as a tunnel port. You assign the tunnel port interface to an access VLAN ID that is unique to each customer.

Figure 26: IEEE 802.1Q Tunnel Ports in a Service-Provider Network

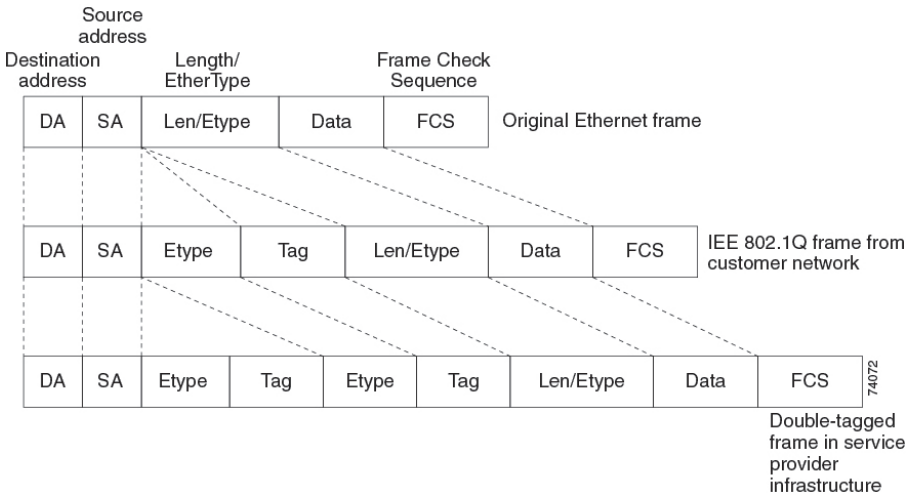


Packets coming from the customer trunk port into the tunnel port on the service-provider edge device are normally IEEE 802.1Q-tagged with the appropriate VLAN ID. The tagged packets remain intact inside the device and when they exit the trunk port into the service-provider network, they are encapsulated with another layer of an IEEE 802.1Q tag (called the metro tag) that contains the VLAN ID that is unique to the customer. The original customer IEEE 802.1Q tag is preserved in the encapsulated packet. Therefore, packets entering the service-provider network are double-tagged, with the outer (metro) tag containing the customer's access VLAN ID, and the inner VLAN ID being that of the incoming traffic.

When the double-tagged packet enters another trunk port in a service-provider core device, the outer tag is stripped as the device processes the packet. When the packet exits another trunk port on the same core device, the same metro tag is again added to the packet.

Figure 27: Original (Normal), IEEE 802.1Q, and Double-Tagged Ethernet Packet Formats

This figure shows the tag structures of the double-tagged packets.



When the packet enters the trunk port of the service-provider egress device, the outer tag is again stripped as the device internally processes the packet. However, the metro tag is not added when the packet is sent out

the tunnel port on the edge devce into the customer network. The packet is sent as a normal IEEE 802.1Q-tagged frame to preserve the original VLAN numbers in the customer network.

In the above network figure, Customer A was assigned VLAN 30, and Customer B was assigned VLAN 40. Packets entering the edge device tunnel ports with IEEE 802.1Q tags are double-tagged when they enter the service-provider network, with the outer tag containing VLAN ID 30 or 40, appropriately, and the inner tag containing the original VLAN number, for example, VLAN 100. Even if both Customers A and B have VLAN 100 in their networks, the traffic remains segregated within the service-provider network because the outer tag is different. Each customer controls its own VLAN numbering space, which is independent of the VLAN numbering space used by other customers and the VLAN numbering space used by the service-provider network.

At the outbound tunnel port, the original VLAN numbers on the customer's network are recovered. It is possible to have multiple levels of tunneling and tagging, but the device supports only one level in this release.

If traffic coming from a customer network is not tagged (native VLAN frames), these packets are bridged or routed as normal packets. All packets entering the service-provider network through a tunnel port on an edge device are treated as untagged packets, whether they are untagged or already tagged with IEEE 802.1Q headers. The packets are encapsulated with the metro tag VLAN ID (set to the access VLAN of the tunnel port) when they are sent through the service-provider network on an IEEE 802.1Q trunk port. The priority field on the metro tag is set to the interface class of service (CoS) priority configured on the tunnel port. (The default is zero if none is configured.)

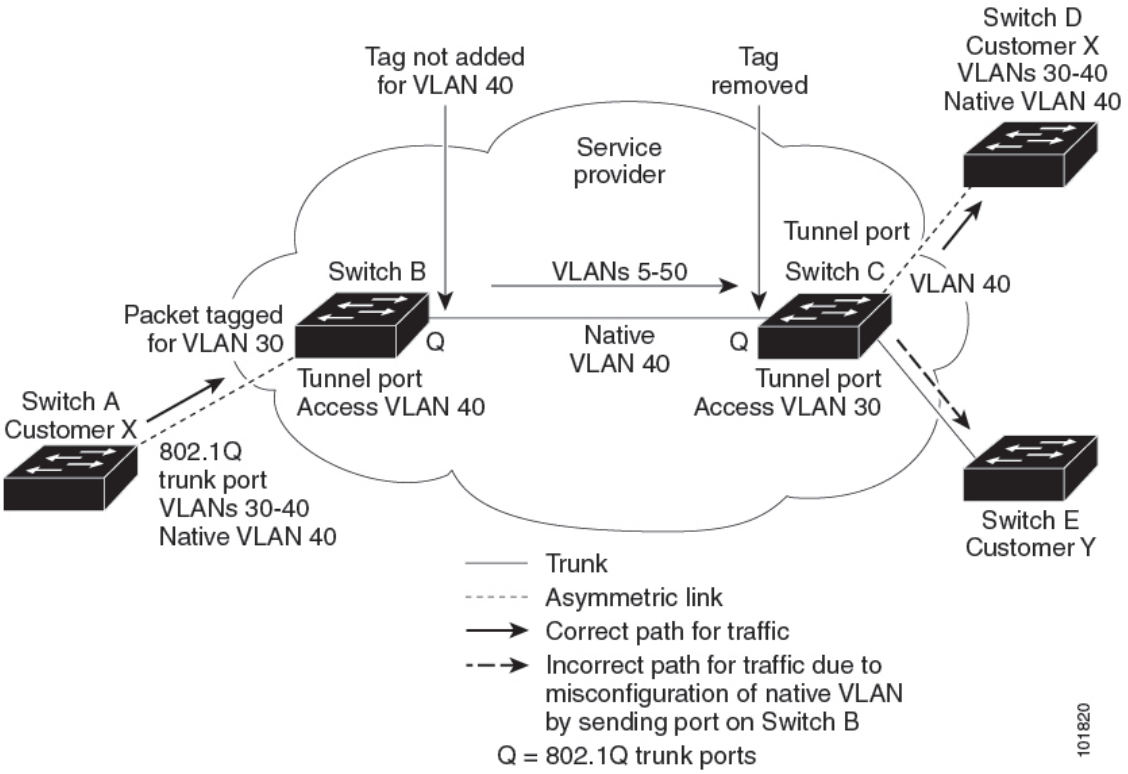
On switches, because 802.1Q tunneling is configured on a per-port basis, it does not matter whether the switch is a standalone device or a member switch. All configuration is done on the active switch.

Native VLANs

When configuring IEEE 802.1Q tunneling on an edge device, you must use IEEE 802.1Q trunk ports for sending packets into the service-provider network. However, packets going through the core of the service-provider network can be carried through IEEE 802.1Q trunks, ISL trunks, or nontrunking links. When IEEE 802.1Q trunks are used in these core devices, the native VLANs of the IEEE 802.1Q trunks must not match any native VLAN of the nontrunking (tunneling) port on the same device because traffic on the native VLAN would not be tagged on the IEEE 802.1Q sending trunk port.

In the following network figure, VLAN 40 is configured as the native VLAN for the IEEE 802.1Q trunk port from Customer X at the ingress edge switch in the service-provider network (Switch B). Switch A of Customer X sends a tagged packet on VLAN 30 to the ingress tunnel port of Switch B in the service-provider network, which belongs to access VLAN 40. Because the access VLAN of the tunnel port (VLAN 40) is the same as the native VLAN of the edge switch trunk port (VLAN 40), the metro tag is not added to tagged packets received from the tunnel port. The packet carries only the VLAN 30 tag through the service-provider network to the trunk port of the egress-edge switch (Switch C) and is misdirected through the egress switch tunnel port to Customer Y.

Figure 28: Potential Problems with IEEE 802.1Q Tunneling and Native VLANs



These are some ways to solve this problem:

- Use the **vlan dot1q tag native** global configuration command to configure the edge switches so that all packets going out an IEEE 802.1Q trunk, including the native VLAN, are tagged. If the switch is configured to tag native VLAN packets on all IEEE 802.1Q trunks, the switch drops untagged packets, and sends and receives only tagged packets.
- Ensure that the native VLAN ID on the edge switches trunk port is not within the customer VLAN range. For example, if the trunk port carries traffic of VLANs 100 to 200, assign the native VLAN a number outside that range.

System MTU

The default system MTU for traffic on the device is 1500 bytes.

You can configure 10-Gigabit and Gigabit Ethernet ports to support frames larger than 1500 bytes by using the **system mtu** *bytes* global configuration command.

The system MTU and system jumbo MTU values do not include the IEEE 802.1Q header. Because the IEEE 802.1Q tunneling feature increases the frame size by 4 bytes when the metro tag is added, you must configure all devices in the service-provider network to be able to process maximum frames by adding 4 bytes to the system MTU size.

For example, the device supports a maximum frame size of 1496 bytes with this configuration: The device has a system MTU value of 1500 bytes, and the **switchport mode dot1q tunnel** interface configuration command is configured on a 10-Gigabit or Gigabit Ethernet device port.

IEEE 802.1Q Tunneling and Other Features

Although IEEE 802.1Q tunneling works well for Layer 2 packet switching, there are incompatibilities between some Layer 2 features and Layer 3 switching.

- A tunnel port cannot be a routed port.
- IP routing is not supported on a VLAN that includes IEEE 802.1Q tunnel ports. Packets that are received from a tunnel port are forwarded based only on Layer 2 information. If routing is enabled on a switch virtual interface (SVI) that includes tunnel ports, untagged IP packets received from the tunnel port are recognized and routed by the switch. Customers can access the Internet through its native VLAN. If this access is not needed, you should not configure SVIs on VLANs that include tunnel ports.
- Fallback bridging is not supported on tunnel ports. Because all IEEE 802.1Q-tagged packets that are received from a tunnel port are treated as non-IP packets, if fallback bridging is enabled on VLANs that have tunnel ports that are configured, IP packets would be improperly bridged across VLANs. Therefore, you must not enable fallback bridging on VLANs with tunnel ports.
- Tunnel ports do not support IP access control lists (ACLs).
- Layer 3 quality of service (QoS) ACLs and other QoS features related to Layer 3 information are not supported on tunnel ports. MAC-based QoS is supported on tunnel ports.
- EtherChannel port groups are compatible with tunnel ports as long as the IEEE 802.1Q configuration is consistent within an EtherChannel port group.
- Port Aggregation Protocol (PAgP), Link Aggregation Control Protocol (LACP), and UniDirectional Link Detection (UDLD) are supported on IEEE 802.1Q tunnel ports.
- Dynamic Trunking Protocol (DTP) is not compatible with IEEE 802.1Q tunneling because you must manually configure asymmetric links with tunnel ports and trunk ports.
- VLAN Trunking Protocol (VTP) does not work between devices that are connected by an asymmetrical link or devices that communicate through a tunnel.
- Loopback detection is supported on IEEE 802.1Q tunnel ports.
- When a port is configured as an IEEE 802.1Q tunnel port, spanning-tree bridge protocol data unit (BPDU) filtering is automatically enabled on the interface. Cisco Discovery Protocol (CDP) is automatically disabled on the interface.

Note When you are configuring IEEE 802.1Q tunneling, the BPDU filtering configuration information is not displayed as spanning-tree BPDU filter is automatically enabled. You can verify the BPDU filter information using the **show spanning tree interface** command.

- When an IEEE 802.1Q tunnel port is configured as SPAN source, span filter must be applied for SVLAN to avoid packet loss.
- IGMP/MLD packet forwarding can be enabled on IEEE 802.1Q tunnels. This can be done by disabling IGMP/MLD snooping on the service provider network.

Default IEEE 802.1Q Tunneling Configuration

By default, IEEE 802.1Q tunneling is disabled because the default switchport mode is dynamic auto. Tagging of IEEE 802.1Q native VLAN packets on all IEEE 802.1Q trunk ports is also disabled.

How to Configure IEEE 802.1Q Tunneling

Follow these steps to configure a port as an IEEE 802.1Q tunnel port:

Before you begin

- Always use an asymmetrical link between the customer device and the edge device, with the customer device port configured as an IEEE 802.1Q trunk port and the edge device port configured as a tunnel port.
- Assign tunnel ports only to VLANs that are used for tunneling.
- Observe configuration requirements for native VLANs and for and maximum transmission units (MTUs).

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Enters interface configuration mode for the interface to be configured as a tunnel port. This should be the edge port in the service-provider network that connects to the customer device. Valid interfaces include physical interfaces and port-channel logical interfaces (port channels 1 to 48).
Step 4	**switchport access vlan** *vlan-id* **Example:** `Device(config-if)# switchport access vlan 2`	Specifies the default VLAN, which is used if the interface stops trunking. This VLAN ID is specific to the particular customer.
Step 5	**switchport mode dot1q-tunnel** **Example:** `Device(config-if)# switchport mode dot1q-tunnel`	Sets the interface as an IEEE 802.1Q tunnel port. **Note**

	Command or Action	Purpose
		Use the **no switchport mode dot1q-tunnel** interface configuration command to return the port to the default state of dynamic desirable.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 7	**vlan dot1q tag native** **Example:** `Device(config)# vlan dot1q tag native`	(Optional) Sets the device to enable tagging of native VLAN packets on all IEEE 802.1Q trunk ports. When not set, and a customer VLAN ID is the same as the native VLAN, the trunk port does not apply a metro tag, and packets could be sent to the wrong destination. **Note** Use the **no vlan dot1q tag native** global configuration command to disable tagging of native VLAN packets.
Step 8	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 9	Use one of the following: • **show dot1q-tunnel** • **show running-config interface** **Example:** `Device# show dot1q-tunnel` or `Device# show running-config interface`	Displays the ports that are configured for IEEE 802.1Q tunneling. Displays the ports that are in tunnel mode.
Step 10	**show vlan dot1q tag native** **Example:** `Device# show vlan dot1q native`	Displays IEEE 802.1Q native VLAN tagging status.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring Tunneling Status

The following table describes the commands used to monitor tunneling status.

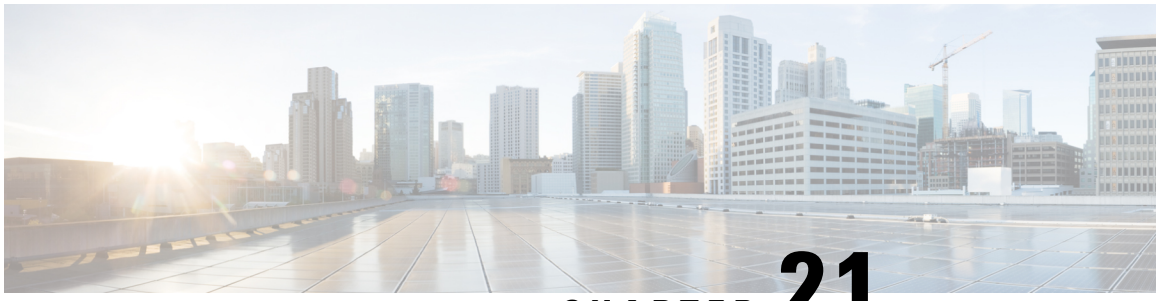
Table 30: Commands for Monitoring Tunneling

Command	Purpose
show dot1q-tunnel	Displays IEEE 802.1Q tunnel ports on the device.
show dot1q-tunnel interface *interface-id*	Verifies if a specific interface is a tunnel port.
show vlan dot1q tag native	Displays the status of native VLAN tagging on the device.

Example: Configuring an IEEE 802.1Q Tunneling Port

The following example shows how to configure an interface as a tunnel port, enable tagging of native VLAN packets, and verify the configuration.

```
Device(config)# interface gigabitethernet 1/1
Device(config-if)# switchport access vlan 22
% Access VLAN does not exist. Creating vlan 22
Device(config-if)# switchport mode dot1q-tunnel
Device(config-if)# exit
Device(config)# vlan dot1q tag native
Device(config)# end
Device# show dot1q-tunnel interface gigabitethernet1/1
Port
-----
Gi1/1Port
-----
Device# show vlan dot1q tag native
dot1q native vlan tagging is enabled
```



CHAPTER 21

Configuring VLAN Mapping

Prerequisites for VLAN Mapping

- By default, no VLAN mapping is configured.
- To process control traffic consistently, either enable Layer 2 protocol tunneling (recommended), as follows:

```
!
Device(config)# interface GigabitEthernet1/1
Device(config-if)# switchport mode access
Device(config-if)# l2protocol-tunnel stp
Device(config-if)# end
```

or insert a BPDU filter for spanning tree, as follows:

```
!
Device(config)# interface GigabitEthernet1/1
Device(config-if)# switchport mode trunk
Device(config-if)# switchport vlan mapping 10 20
Device(config-if)# spanning-tree bpdufilter enable
Device(config-if)# end
```

Prerequisites for One to One VLAN Mapping

- One-to-One VLAN mapping can be configured only on trunk ports and not on dynamic trunk.
- One-to-One VLAN mapping should be identical on both ports.
- S-VLAN should be created and present in the allowed VLAN list of the trunk port where One-to-One VLAN mapping is configured.

Restrictions for VLAN Mapping

- If VLAN mapping is enabled on an EtherChannel, the configuration does not apply to all member ports of the EtherChannel bundle but applies only to the EtherChannel interface.
- If VLAN mapping is enabled on an EtherChannel and a conflicting mapping translation is enabled on a member port, the port is removed from the EtherChannel.
- If a port belonging to an EtherChannel is configured with a VLAN mapping and the EtherChannel is configured with a conflicting VLAN mapping, the port is removed from the EtherChannel.
- The member port of an EtherChannel is removed from the EtherChannel bundle if the mode of the port is changed to anything other than 'trunk' mode.
- Default native VLANs, user-configured native VLANs, and reserved VLANs cannot be used for VLAN mapping.
- The S-VLAN used for VLAN mapping cannot be a part of any other Layer 3 configurations, EVPN, or LISP.
- PVLAN support is not available when VLAN mapping is configured.

Restrictions for One to One VLAN Mapping

- When One-to-One VLAN mapping is configured, multiple C-VLANs cannot be mapped to the same S-VLAN
- Merging of C-VLAN and S-VLAN spanning-tree topology is not supported in case of one-to-one vlan mapping.

About VLAN Mapping

In a typical deployment of VLAN mapping, you want service provider to provide a transparent switching infrastructure that includes customers' switches at the remote location as a part of local site. This allows customers to use the same VLAN ID space and run Layer 2 control protocols seamlessly across the provider network. In such scenarios, we recommend that service providers do not impose their VLAN IDs on their customers.

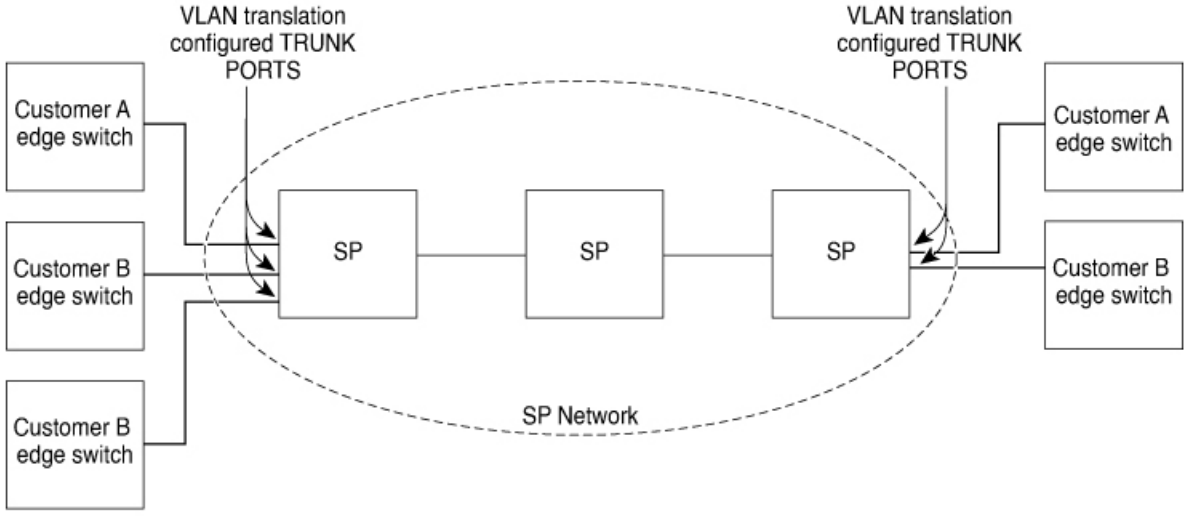
One way to establish translated VLAN IDs (S-VLANs) is to map customer VLAN to Service Provider VLAN on trunk ports that are connected to a customer network. Packets entering the port are mapped to service provider VLAN (S-VLAN) based on the port number and the packet's original customer VLAN-ID (C-VLAN).

Service providers' internal assignments might conflict with a customer's VLAN. To isolate customer traffic, a service provider decides to map a specific VLAN into another one while the traffic is in its cloud.

Deployment Example

In the figure, the service provider provides Layer 2 VPN service to two different customers, A and B. The service provider separates the data and control traffic between the two customers and from the providers' own control traffic. The service provider network must also be transparent to the customer edge devices.

Figure 29: Example of a Service Provider with Layer 2 VPN Service



All forwarding operations on IE3500 series switch are performed using S-VLAN and not C-VLAN information because the VLAN ID is mapped to the S-VLAN on ingress.

Note When you configure features on a port for VLAN mapping, you always use the S-VLAN rather than C-VLAN.

On an interface configured for VLAN mapping, the specified C-VLAN packets are mapped to the specified S-VLAN when they enter the port. Symmetrical mapping to the customer C-VLAN occurs when packets exit the port.

The switch supports these types of VLAN mapping on trunk ports:

- One-to-one VLAN mapping.
- Selective QinQ.
- QinQ on a trunk port.

Figure 30: Mapping Customer VLANs to Service-Provider VLANs

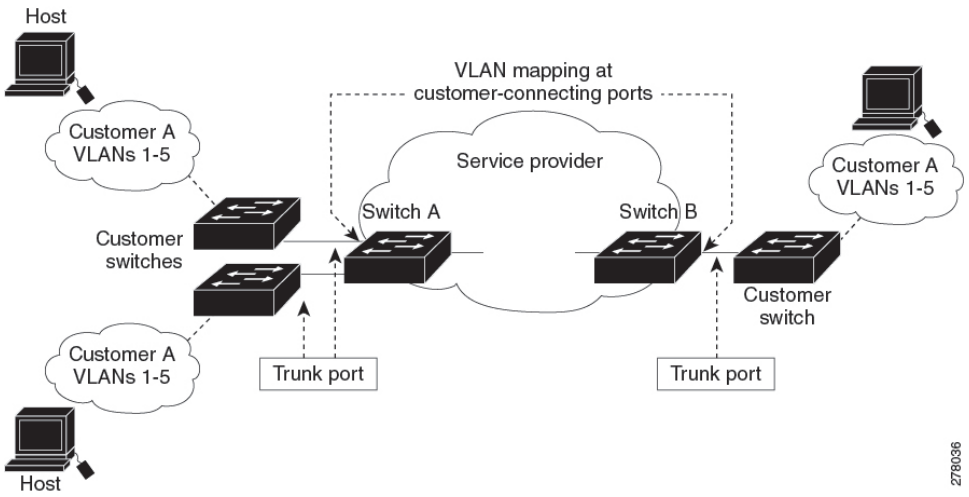


Figure shows a topology where a customer uses the same VLANs in multiple sites on different sides of a service-provider network. The C-VLAN IDs is mapped to service-provider VLAN IDs for packet travel across the service-provider backbone. The C-VLAN IDs are retrieved at the other side of the service-provider backbone for use in the other customer site. Configure the same set of VLAN mappings at a customer-connected port on each side of the service-provider network.

One-to-One VLAN Mapping

One-to-one VLAN mapping occurs at the ingress and egress of the port and maps the customer C-VLAN ID in the 802.1Q tag to the service-provider S-VLAN ID. You can also specify that packets with all other Vlan IDs are forwarded.

Selective Q-in-Q

Selective QinQ maps the specified customer VLANs entering the UNI to the specified S-VLAN ID. The S-VLAN ID is added to the incoming unmodified C-VLAN and the packet travels the service provider network double-tagged. At the egress, the S-VLAN ID is removed and the customer VLAN-ID is retained on the packet. By default, packets that do not match the specified customer VLANs are dropped.

Q-in-Q on a Trunk Port

QinQ on a trunk port maps all the customer VLANs entering the UNI to the specified S-VLAN ID. Similar to Selective QinQ, the packet is double-tagged and at the egress, the S-VLAN ID is removed.

Configuration Guidelines for VLAN Mapping

Note

- By default, no VLAN mapping is configured.
- Maximum number of VLAN mapping configurations supported is 512 system wide.

Guidelines include the following:

- If the VLAN mapping is enabled on an EtherChannel, the configuration does not apply to all member ports of the EtherChannel bundle and applies only to the EtherChannel interface.
- If the VLAN mapping is enabled on an EtherChannel and a conflicting mapping/translation is enabled on a member port, then the port is removed from the EtherChannel.
- If a port belonging to an EtherChannel is configured with a VLAN mapping and the EtherChannel is configured with a conflicting VLAN mapping, then the port is removed from the EtherChannel.
- The member port of an EtherChannel is removed from the EtherChannel bundle if the mode of the port is changed to anything other than 'trunk' mode.
- To process control traffic consistently, either enable Layer 2 protocol tunneling (recommended), as follows:

```
!
Device(config)# interface GigabitEthernet1/1
Device(config-if)# switchport mode trunk
Device(config-if)#switchport vlan mapping 20 300
Device(config-if)# l2protocol-tunnel stp
Device(config-if)# end
```

  or insert a BPDU filter for spanning tree, as follows:

```
!
Device(config)# interface GigabitEthernet1/1
Device(config-if)# switchport mode trunk
Device(config-if)# switchport vlan mapping 10 20
Device(config-if)# spanning-tree bpdufilter enable
Device(config-if)# end
```

- Default native VLANs, user-configured native VLANs, and reserved VLANs (range 1002-1005) cannot be used for VLAN mapping.
- The S-VLAN used for VLAN mapping cannot be a part of any other Layer 3 configurations like EVPN or LISP.
- PVLAN support is not available when VLAN mapping is configured.

Configuration Guidelines for One-to-One VLAN Mapping

- One-to-One VLAN mapping can be configured only on trunk ports and not on dynamic trunk.
- One-to-One VLAN mapping should be identical on both ports.

- S-VLAN should be created and present in the allowed VLAN list of the trunk port where One-to-One VLAN mapping is configured.
- When One-to-One VLAN mapping is configured, multiple C-VLANs cannot be mapped to the same S-VLAN.
- Merging of C-VLAN and S-VLAN spanning-tree topology is not supported in case of one-to-one VLAN mapping.

Configuration Guidelines for Selective Q-in-Q

- S-VLAN should be created and present in the allowed VLAN list of the trunk port where Selective Q-in-Q is configured.
- When Selective Q-in-Q is configured, the device supports Layer 2 protocol tunneling for CDP, STP, LLDP, and VTP. For emulated point-to-point network topologies, it also supports PAgP, LACP, and UDLD protocols.
- IP routing is not supported on Selective Q-in-Q enabled ports.
- IPSG is not supported on Selective Q-in-Q enabled ports.

Configuration Guidelines for Q-in-Q on a Trunk Port

- S-VLAN should be created and present in the allowed VLAN list of the trunk port where Q-in-Q on a trunk port is configured.
- When Q-in-Q on a trunk port is configured, the device supports Layer 2 protocol tunneling for CDP, STP, LLDP, and VTP. For emulated point-to-point network topologies, it also supports PAgP, LACP, and UDLD protocols.
- Ingress and egress SPAN, and RSPAN are supported on trunk ports with QinQ enabled.
- When QinQ is enabled, the SPAN filtering can be enabled to monitor only the traffic on the mapped VLAN, i.e. S-VLANs.
- IGMP snooping is not supported on the C-VLAN.

How to Configure VLAN Mapping

The following sections provide information about configuring VLAN mapping:

One-to-One VLAN Mapping

Note VLAN Mapping is supported only with the **network-essentials** license level.

To configure one-to-one VLAN mapping to map a customer VLAN ID to a service-provider VLAN ID, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Enters interface configuration mode for the interface that is connected to the service-provider network. You can enter a physical interface or an EtherChannel port channel.
Step 4	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Configures the interface as a trunk port.
Step 5	**switchport vlan mapping** *vlan-id translated-id* **Example:** `Device(config-if)# switchport vlan mapping 2 102`	Enters the VLAN IDs to be mapped: • vlan-id —the customer VLAN ID (C-VLAN) entering the switch from the customer network. The range is from 1 to 4094. • translated-id —the assigned service-provider VLAN ID (S-VLAN). The range is from 1 to 4094.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 7	**spanning-tree bpdufilter enable** **Example:** `Device(config)# spanning-tree bpdufilter enable`	Inserts a BPDU filter for spanning tree. **Note** To process control traffic consistently, either enable Layer 2 protocol tunneling (recommended) or insert a BPDU filter for spanning tree.
Step 8	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 9	**show vlan mapping** **Example:** `Device# show vlan mapping`	Verifies the configuration.
Step 10	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Example

Use **no switchport vlan mapping** command to remove the VLAN mapping information. Entering **no switchport vlan mapping all** command deletes all mapping configurations.

This example shows how to map VLAN IDs 2 to 6 in the customer network to VLANs 101 to 105 in the service-provider network (Figure 3-5). You configure the same VLAN mapping commands for a port in Switch A and Switch B; the traffic on all other VLAN IDs is forwarded as normal traffic.

```
Device> enable
Device# configure terminal
Device(config)# interface gigabiethernet1/1
Device(config-if)# switchport vlan mapping 2 101
Device(config-if)# switchport vlan mapping 3 102
Device(config-if)# switchport vlan mapping 4 103
Device(config-if)# switchport vlan mapping 5 104
Device(config-if)# switchport vlan mapping 6 105
Device(config-if)# exit
```

In the previous example, at the ingress of the service-provider network, VLAN IDs 2 to 6 in the customer network are mapped to VLANs 101 to 105, in the service provider network. At the egress of the service provider network, VLANs 101 to 105 in the service provider network are mapped to VLAN IDs 2 to 6, in the customer network.

Note Packets with VLAN IDs other than the ones with configured VLAN Mapping are forwarded as normal traffic.

Use **show vlan mapping** command to view information about configured vlans.

```
Device> enable
Device# configure terminal
Device(config)# show vlan mapping
Total no of vlan mappings configured: 1
Interface Po5:
VLANs on wire                  Translated     VLAN Operation
------------------------------ -------------- --------------
20                               30            1-to-1
```

Selective Q-in-Q on a Trunk Port

To configure VLAN mapping for selective Q-in-Q on a trunk port, perform this task:

Note You cannot configure one-to-one mapping and selective Q-in-Q on the same interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Enters interface configuration mode for the interface that is connected to the service-provider network. You can enter a physical interface or an EtherChannel port channel.
Step 4	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Configures the interface as a trunk port.
Step 5	**switchport vlan mapping** *vlan-id* **dot1q-tunnel** *outer vlan-id* **Example:** `Device(config-if)# switchport vlan mapping 16 dot1q-tunnel 64`	Enters the VLAN IDs to be mapped: • vlan-id —the customer VLAN ID (C-VLAN) entering the switch from the customer network. The range is from 1 to 4094. You can enter a string of VLAN-IDs. • outer-vlan-id —The outer VLAN ID (S-VLAN) of the service provider network. The range is from 1 to 4094. Use the no form of this command to remove the VLAN mapping configuration. Entering the **no switchport vlan mapping all** command deletes all mapping configurations.
Step 6	**switchport vlan mapping default dot1q-tunnel** *vlan-id* **Example:**	Specifies that all unmapped packets on the port are forwarded with the specified S-VLAN.

	Command or Action	Purpose
	`Device(config-if)# switchport vlan mapping default dot1q-tunnel 22`	By default, packets that do not match the mapped VLANs, are dropped. Untagged traffic are forwarded without dropping.
Step 7	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 8	**spanning-tree bpdufilter enable** **Example:** `Device(config)# spanning-tree bpdufilter enable`	Inserts a BPDU filter for spanning tree. **Note** To process control traffic consistently, either enable Layer 2 protocol tunneling (recommended) or insert a BPDU filter for spanning tree.
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 10	**show interfaces** *interface-id* **vlan mapping** **Example:** `Device# show interfaces gigabitethernet1/1 vlan mapping`	Verifies the configuration.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Example

This example shows how to configure selective QinQ mapping on the port so that traffic with a C-VLAN ID of 2 to 5 enters the switch with an S-VLAN ID of 100. By default, the traffic of any other VLAN ID is dropped.

```
Device(config)# interface GigabitEthernet1/1
Device(config-if)# switchport vlan mapping 2-5 dot1q-tunnel 100
Device(config-if)# exit
```

This example shows how to configure selective QinQ mapping on the port so that traffic with a C-VLAN ID of 2 to 5 enters the switch with an S-VLAN ID of 100. The traffic of any other VLAN ID is forwarded with the S-VLAN ID of 200.

```
Device(config)# interface GigabiEthernet1/1
Device(config-if)# switchport vlan mapping 2-5 dot1q-tunnel 100
Device(config-if)# switchport vlan mapping default dot1q-tunnel 200
Device(config-if)# exit
```

```
Device# show vlan mapping
Total no of vlan mappings configured: 5
Interface gi1/1:
VLANs on wire                   Translated VLAN     Operation
------------------------------  ---------------     --------------
2-5                                  100            selective QinQ
*                                    200            default QinQ
```

Q-in-Q on a Trunk Port

To configure VLAN mapping for Q-in-Q on a trunk port, perform this task:

Procedure

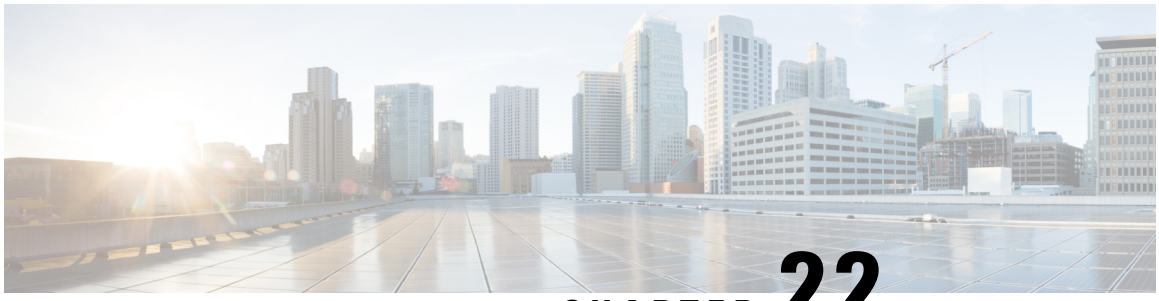
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Enters interface configuration mode for the interface that is connected to the service-provider network. You can enter a physical interface or an EtherChannel port channel.
Step 4	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Configures the interface as a trunk port.
Step 5	**switchport vlan mapping default dot1q-tunnel** *vlan-id* **Example:** `Device(config-if)# switchport vlan mapping default dot1q-tunnel 16`	Specifies that all unmapped C-VLAN packets on the port are forwarded with the specified S-VLAN.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 7	**spanning-tree bpdufilter enable** **Example:** `Device(config)# spanning-tree bpdufilter enable`	Inserts a BPDU filter for spanning tree. **Note** To process control traffic consistently, either enable Layer 2 protocol tunneling

	Command or Action	Purpose
		(recommended) or insert a BPDU filter for spanning tree.
Step 8	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 9	**show interfaces** *interface-id* **vlan mapping** **Example:** `Device# show interfaces gigabitethernet1/1 vlan mapping`	Verifies the configuration.
Step 10	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Example

This example shows how to configure QinQ mapping on the port so that traffic of any VLAN ID is forwarded with the S-VLAN ID of 200.

```
Device(config)# interface gigabiethernet1/1
Device(config-if)# switchport vlan mapping default dot1q-tunnel 200
Device(config-if)# exit
```



CHAPTER 22

Configuring VTP

The following sections provide information about Configuring VTP:

Prerequisites for VTP

Before you create VLANs, you must decide whether to use the VLAN Trunking Protocol (VTP) in your network. Using VTP, you can make configuration changes centrally on one or more devices and have those changes automatically communicated to all the other devices in the network. Without VTP, you cannot send information about VLANs to other devices.

VTP is designed to work in an environment where updates are made on a single device and are sent through VTP to other devices in the domain. It does not work well in a situation where multiple updates to the VLAN database occur simultaneously on devices in the same domain, which would result in an inconsistency in the VLAN database.

You can enable or disable VTP per port by entering the [**no**] **vtp** interface configuration command. When you disable VTP on trunking ports, all VTP instances for that port are disabled. You cannot set VTP to *off* for the MST database and *on* for the VLAN database on the same port.

When you globally set VTP mode to off, it applies to all the trunking ports in the system. However, you can specify on or off on a per-VTP instance basis. For example, you can configure the device as a VTP server for the VLAN database but with VTP *off* for the MST database.

Because trunk ports send and receive VTP advertisements, you must ensure that at least one trunk port is configured on the device and that this trunk port is connected to the trunk port of another device. Otherwise, the device cannot receive any VTP advertisements.

Restrictions for VTP

The following are restrictions for a VTP:

Caution Before adding a VTP client device to a VTP domain, always verify that its VTP configuration revision number is lower than the configuration revision number of the other devices in the VTP domain. Devices in a VTP domain always use the VLAN configuration of the device with the highest VTP configuration revision number. If you add a device that has a revision number higher than the revision number in the VTP domain, it can erase all VLAN information from the VTP server and VTP domain.

Information About VTP

The following sections provide information about VTP and VTP configuration:

VTP

VTP is a Layer 2 messaging protocol that maintains VLAN configuration consistency by managing the addition, deletion, and renaming of VLANs on a network-wide basis. VTP minimizes misconfigurations and configuration inconsistencies that can cause several problems, such as duplicate VLAN names, incorrect VLAN-type specifications, and security violations.

VTP Domain

A VTP domain (also called a VLAN management domain) consists of one device or several interconnected devices under the same administrative responsibility sharing the same VTP domain name. A device can be in only one VTP domain. You make global VLAN configuration changes for the domain.

By default, the device is in the VTP no-management-domain state until it receives an advertisement for a domain over a trunk link (a link that carries the traffic of multiple VLANs) or until you configure a domain name. You can create or modify VLANs on a VTP server without specifying the domain name. However, when the management domain name is not specified VLAN information is not propagated over the network.

If the device receives a VTP advertisement over a trunk link, it inherits the management domain name and the VTP configuration revision number. The device then ignores advertisements with a different domain name or an earlier configuration revision number.

When you make a change to the VLAN configuration on a VTP server, the change is propagated to all devices in the VTP domain. VTP advertisements are sent over all IEEE trunk connections, including IEEE 802.1Q. VTP dynamically maps VLANs with unique names and internal index associates across multiple LAN types. Mapping eliminates excessive device administration required from network administrators.

If you configure a device for VTP transparent mode, you can create and modify VLANs, but the changes are not sent to other devices in the domain, and they affect only the individual device. However, configuration changes made when the device is in this mode are saved in the device running configuration and can be saved to the device startup configuration file.

VTP Modes

Table 31: VTP Modes

VTP Mode	Description
VTP server	In VTP server mode, you can create, modify, and delete VLANs, and specify other configuration parameters (such as the VTP version) for the entire VTP domain. VTP servers advertise their VLAN configurations to other devices in the same VTP domain and synchronize their VLAN configurations with other devices based on advertisements received over trunk links. VTP server is the default mode. In VTP server mode, VLAN configurations are saved in NVRAM. If the device detects a failure while writing a configuration to NVRAM, VTP mode automatically changes from server mode to client mode. If this happens, the device cannot be returned to VTP server mode until the NVRAM is functioning.
VTP client	A VTP client functions like a VTP server and transmits and receives VTP updates on its trunks, but you cannot create, change, or delete VLANs on a VTP client. VLANs are configured on another device in the domain that is in server mode. In VTP versions 1 and 2 in VTP client mode, VLAN configurations are not saved in NVRAM. In VTP version 3, VLAN configurations are saved in NVRAM in client mode.
VTP transparent	VTP transparent devices do not participate in VTP. A VTP transparent device does not advertise its VLAN configuration and does not synchronize its VLAN configuration based on received advertisements. However, in VTP version 2 or version 3, transparent devices do forward VTP advertisements that they receive from other devices through their trunk interfaces. You can create, modify, and delete VLANs on a device in VTP transparent mode. In VTP versions 1 and 2, the device must be in VTP transparent mode when you create private VLANs and when they are configured, you should not change the VTP mode from transparent to client or server mode. VTP version 3 also supports private VLANs in client and server modes. When private VLANs are configured, do not change the VTP mode from transparent to client or server mode. When the device is in VTP transparent mode, the VTP and VLAN configurations are saved in NVRAM, but they are not advertised to other devices. In this mode, VTP mode and domain name are saved in the device running configuration, and you can save this information in the device startup configuration file by using the **copy running-config startup-config** privileged EXEC command.

VTP Mode	Description
VTP off	A device in VTP off mode functions in the same manner as a VTP transparent device, except that it does not forward VTP advertisements on trunks.

VTP Advertisements

Each device in the VTP domain sends periodic global configuration advertisements from each trunk port to a reserved multicast address. Neighboring devices receive these advertisements and update their VTP and VLAN configurations as necessary.

VTP advertisements distribute this global domain information:

- VTP domain name
- VTP configuration revision number
- Update identity and update timestamp
- MD5 digest VLAN configuration, including maximum transmission unit (MTU) size for each VLAN
- Frame format

VTP advertisements distribute this VLAN information for each configured VLAN:

- VLAN IDs (including IEEE 802.1Q)
- VLAN name
- VLAN type
- VLAN state
- Additional VLAN configuration information specific to the VLAN type

In VTP version 3, VTP advertisements also include the primary server ID, an instance number, and a start index.

VTP Version 2

If you use VTP in your network, you must decide which version of VTP to use. By default, VTP operates in version 1.

VTP version 2 supports these features that are not supported in version 1:

- Token Ring support—VTP version 2 supports Token Ring Bridge Relay Function (TrBRF) and Token Ring Concentrator Relay Function (TrCRF) VLANs.
- Unrecognized Type-Length-Value (TLV) support—A VTP server or client propagates configuration changes to its other trunks, even for TLVs it is not able to parse. The unrecognized TLV is saved in NVRAM when the device is operating in VTP server mode.
- Version-Dependent Transparent Mode—In VTP version 1, a VTP transparent device inspects VTP messages for the domain name and version and forwards a message only if the version and domain name

match. Although VTP version 2 supports only one domain, a VTP version 2 transparent device forwards a message only when the domain name matches.

- Consistency Checks—In VTP version 2, VLAN consistency checks (such as VLAN names and values) are performed only when you enter new information through the CLI or SNMP. Consistency checks are not performed when new information is obtained from a VTP message or when information is read from NVRAM. If the MD5 digest on a received VTP message is correct, its information is accepted.

VTP Version 3

VTP version 3 supports these features that are not supported in version 1 or version 2:

- Enhanced authentication—You can configure the authentication as **hidden** or **secret**. When **hidden**, the secret key from the password string is saved in the VLAN database file, but it does not appear in plain text in the configuration. Instead, the key associated with the password is saved in hexadecimal format in the running configuration. You must reenter the password if you enter a takeover command in the domain. When you enter the **secret** keyword, you can directly configure the password secret key.
- Support for extended range VLAN (VLANs 1006 to 4094) database propagation—VTP versions 1 and 2 propagate only VLANs 1 to 1005.

Note VTP pruning still applies only to VLANs 1 to 1005, and VLANs 1002 to 1005 are still reserved and cannot be modified.

- Private VLAN support.
- Support for any database in a domain—In addition to propagating VTP information, version 3 can propagate Multiple Spanning Tree (MST) protocol database information. A separate instance of the VTP protocol runs for each application that uses VTP.
- VTP primary server and VTP secondary servers—A VTP primary server updates the database information and sends updates that are honored by all devices in the system. A VTP secondary server can only back up the updated VTP configurations received from the primary server to its NVRAM.

  By default, all devices come up as secondary servers. You can enter the **vtp primary** privileged EXEC command to specify a primary server. Primary server status is only needed for database updates when the administrator issues a takeover message in the domain. You can have a working VTP domain without any primary servers. Primary server status is lost if the device reloads, after a switchover, or domain parameters change, even when a password is configured on the device.

VTP Pruning

VTP pruning increases network available bandwidth by restricting flooded traffic to those trunk links that the traffic must use to reach the destination devices. Without VTP pruning, a device floods broadcast, multicast, and unknown unicast traffic across all trunk links within a VTP domain even though receiving devices might discard them. VTP pruning is disabled by default.

VTP pruning blocks unneeded flooded traffic to VLANs on trunk ports that are included in the pruning-eligible list. Only VLANs included in the pruning-eligible list can be pruned. By default, VLANs 2 through 1001 are

pruning eligible device trunk ports. If the VLANs are configured as pruning-ineligible, the flooding continues. VTP pruning is supported in all VTP versions.

Figure 31: Flooding Traffic without VTP Pruning

VTP pruning is disabled in the switched network. Port 1 on Device A and Port 2 on Device D are assigned to the Red VLAN. If a broadcast is sent from the host connected to Device A, Device A floods the broadcast and every device in the network receives it, even though Devices C, E, and F have no ports in the Red VLAN.

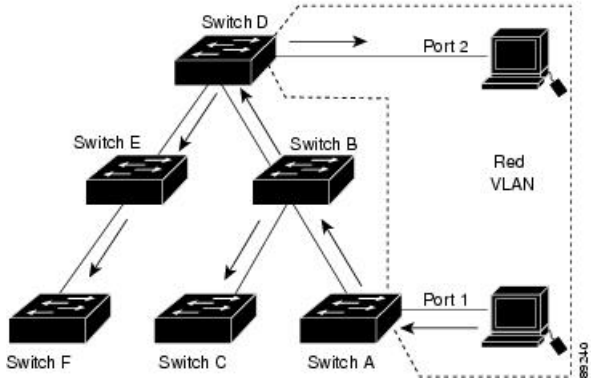
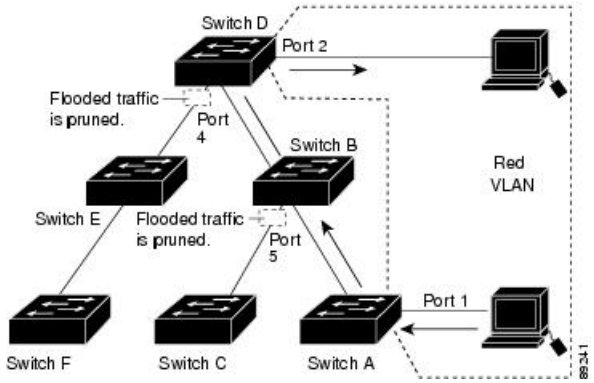


Figure 32: Optimized Flooded Traffic VTP Pruning

VTP pruning is enabled in the switched network. The broadcast traffic from Device A is not forwarded to Devices C, E, and F because traffic for the Red VLAN has been pruned on the links shown (Port 5 on Device B and Port 4 on Device D).



With VTP versions 1 and 2, when you enable pruning on the VTP server, it is enabled for the entire VTP domain. In VTP version 3, you must manually enable pruning on each device in the domain. Making VLANs pruning-eligible or pruning-ineligible affects pruning eligibility for those VLANs on that trunk only (not on all devices in the VTP domain).

VTP pruning takes effect several seconds after you enable it. VTP pruning does not prune traffic from VLANs that are pruning-ineligible. VLAN 1 and VLANs 1002 to 1005 are always pruning-ineligible; traffic from these VLANs cannot be pruned. Extended-range VLANs (VLAN IDs higher than 1005) are also pruning-ineligible.

VTP Configuration Guidelines

This section provides information about VTP configuration guidelines:

VTP Configuration Requirements

When you configure VTP, you must configure a trunk port so that the device can send and receive VTP advertisements to and from other devices in the domain.

VTP versions 1 and 2 do not support private VLANs. VTP version 3 does support private VLANs. If you configure private VLANs, the device must be in VTP transparent mode. When private VLANs are configured on the device, do not change the VTP mode from transparent to client or server mode.

VTP Settings

The VTP information is saved in the VTP VLAN database. When VTP mode is transparent, the VTP domain name and mode are also saved in the device running configuration file, and you can save it in the device startup configuration file by entering the **copy running-config startup-config** privileged EXEC command. You must use this command if you want to save VTP mode as transparent, even if the device resets.

When you save VTP information in the device startup configuration file and reboot the device, the device configuration is selected as follows:

- If the VTP mode is transparent in the startup configuration and the VLAN database and the VTP domain name from the VLAN database matches that in the startup configuration file, the VLAN database is ignored (cleared), and the VTP and VLAN configurations in the startup configuration file are used. The VLAN database revision number remains unchanged in the VLAN database.
- If the VTP mode or domain name in the startup configuration do not match the VLAN database, the domain name and VTP mode and configuration for VLAN IDs 1 to 1005 use the VLAN database information.

Domain Names for Configuring VTP

When configuring VTP for the first time, you must always assign a domain name. You must configure all devices in the VTP domain with the same domain name. Devices in VTP transparent mode do not exchange VTP messages with other devices, and you do not need to configure a VTP domain name for them.

Note If the NVRAM and DRAM storage is sufficient, all devices in a VTP domain should be in VTP server mode.

Caution Do not configure a VTP domain if all devices are operating in VTP client mode. If you configure the domain, it is impossible to make changes to the VLAN configuration of that domain. Make sure that you configure at least one device in the VTP domain for VTP server mode.

Passwords for the VTP Domain

You can configure a password for the VTP domain, but it is not required. If you do configure a domain password, all domain devices must share the same password and you must configure the password on each device in the management domain. Devices without a password or with the wrong password reject VTP advertisements.

If you configure a VTP password for a domain, a device that is booted without a VTP configuration does not accept VTP advertisements until you configure it with the correct password. After the configuration, the device accepts the next VTP advertisement that uses the same password and domain name in the advertisement.

If you are adding a new device to an existing network with VTP capability, the new device learns the domain name only after the applicable password has been configured on it.

Caution When you configure a VTP domain password, the management domain does not function properly if you do not assign a management domain password to each device in the domain.

VTP Version

Follow these guidelines when deciding which VTP version to implement:

- All devices in a VTP domain must have the same domain name, but they do not need to run the same VTP version.
- A VTP version 2-capable device can operate in the same VTP domain as a device running VTP version 1 if version 2 is disabled on the version 2-capable device (version 2 is disabled by default).
- If a device running VTP version 1, but capable of running VTP version 2, receives VTP version 3 advertisements, it automatically moves to VTP version 2.
- If a device running VTP version 3 is connected to a device running VTP version 1, the VTP version 1 device moves to VTP version 2, and the VTP version 3 device sends scaled-down versions of the VTP packets so that the VTP version 2 device can update its database.
- A device running VTP version 3 cannot move to version 1 or 2 if it has extended VLANs.
- Do not enable VTP version 2 on a device unless all of the devices in the same VTP domain are version-2-capable. When you enable version 2 on a device, all of the version-2-capable devices in the domain enable version 2. If there is a version 1-only device, it does not exchange VTP information with device that have version 2 enabled.
- Cisco recommends placing VTP version 1 and 2 device at the edge of the network because they do not forward VTP version 3 advertisements.
- If there are TrBRF and TrCRF Token Ring networks in your environment, you must enable VTP version 2 or version 3 for Token Ring VLAN switching to function properly. To run Token Ring and Token Ring-Net, disable VTP version 2.
- For VTP version 1 and version 2, the device must be in VTP transparent mode when you create extended-range VLANs. VTP version 3 also supports creating extended-range VLANs in client or server mode.
- When a VTP version 3 device trunk port receives messages from a VTP version 2 device, it sends a scaled-down version of the VLAN database on that particular trunk in VTP version 2 format. A VTP version 3 device does not send VTP version 2-formatted packets on a trunk unless it first receives VTP version 2 packets on that trunk port.
- When a VTP version 3 device detects a VTP version 2 device on a trunk port, it continues to send VTP version 3 packets, in addition to VTP version 2 packets, to allow both kinds of neighbors to coexist on the same trunk.

- A VTP version 3 device does not accept configuration information from a VTP version 2 or version 1 device.
- Two VTP version 3 regions can only communicate in transparent mode over a VTP version 1 or version 2 region.
- Devices that are only VTP version 1 capable cannot interoperate with VTP version 3 devices.

How to Configure VTP

The following sections provide information about Configuring VTP:

Configuring VTP Mode

You can configure VTP mode as one of these:

- VTP server mode—In VTP server mode, you can change the VLAN configuration and have it propagated throughout the network.
- VTP client mode—In VTP client mode, you cannot change its VLAN configuration. The client device receives VTP updates from a VTP server in the VTP domain and then modifies its configuration accordingly.
- VTP transparent mode—In VTP transparent mode, VTP is disabled on the device. The device does not send VTP updates and does not act on VTP updates received from other devices. However, a VTP transparent device running VTP version 2 does forward received VTP advertisements on its trunk links.
- VTP off mode—VTP off mode is the same as VTP transparent mode except that VTP advertisements are not forwarded.

When you configure a domain name, it cannot be removed; you can only reassign a device to a different domain.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**vtp domain** *domain-name* **Example:** `Device(config)# vtp domain eng_group`	Configures the VTP administrative-domain name. The name can be 1 to 32 characters. All devices operating in VTP server or client mode under the same administrative responsibility must be configured with the same domain name. This command is optional for modes other than server mode. VTP server mode requires a domain name. If the device has a trunk connection to a VTP domain, the device learns the domain name from the VTP server in the domain. You should configure the VTP domain before configuring other VTP parameters.
Step 4	**vtp mode** {**client** | **server** | **transparent** | **off**} {**vlan** | **mst** | **unknown**} **Example:** `Device(config)# vtp mode server`	Configures the device for VTP mode (client, server, transparent, or off). • **vlan**—The VLAN database is the default if none are configured. • **mst**—The multiple spanning tree (MST) database. • **unknown**—An unknown database type.
Step 5	**vtp password** *password* **Example:** `Device(config)# vtp password mypassword`	(Optional) Sets the password for the VTP domain. The password can be 8 to 64 characters. If you configure a VTP password, the VTP domain does not function properly if you do not assign the same password to each device in the domain.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show vtp status** **Example:** `Device# show vtp status`	Verifies your entries in the *VTP Operating Mode* and the *VTP Domain Name* fields of the display.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves the configuration in the startup configuration file. Only VTP mode and domain name are saved in the device running configuration and can be copied to the startup configuration file.

Configuring a VTP Version 3 Password

You can configure a VTP version 3 password on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vtp version 3** **Example:** `Device(config)# vtp version 3`	Enables VTP version 3 on the device. The default is VTP version 1.
Step 4	**vtp password** *password* [**hidden** | **secret**] **Example:** `Device(config)# vtp password mypassword hidden`	(Optional) Sets the password for the VTP domain. The password can be 8 to 64 characters. • (Optional) **hidden**: Saves the secret key generated from the password string in the `nvram:vlan.dat` file. If you configure a takeover by configuring a VTP primary server, you are prompted to reenter the password. • (Optional) **secret**: Directly configures the password. The secret password must contain 32 hexadecimal characters.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show vtp password** **Example:** `Device# show vtp password`	Verifies whether the VTP password is configured or not.

	Command or Action	Purpose
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a VTP Version 3 Primary Server

When you configure a VTP server as a VTP primary server, the takeover operation starts.

Procedure

	Command or Action	Purpose
Step 1	**vtp version 3** **Example:** `Device(config)# vtp version 3`	Enables VTP version 3 on the device. The default is VTP version 1.
Step 2	**vtp primary** [**vlan** | **mst**] [**force**] **Example:** `Device# vtp primary vlan force`	Changes the operational state of a device from a secondary server (the default) to a primary server and advertises the configuration to the domain. If the device password is configured as **hidden**, you are prompted to reenter the password. • (Optional) **vlan**—Selects the VLAN database as the takeover feature. This is the default. • (Optional) **mst**—Selects the multiple spanning tree (MST) database as the takeover feature. • (Optional) **force**—Overwrites the configuration of any conflicting servers. If you do not enter **force**, you are prompted for confirmation before the takeover.

Enabling the VTP Version

VTP version 2 and version 3 are disabled by default.

- When you enable VTP version 2 on a device , every VTP version 2-capable device in the VTP domain enables version 2. To enable VTP version 3, you must manually configure it on each device.

- With VTP versions 1 and 2, you can configure the version only on devices in VTP server or transparent mode. If a device is running VTP version 3, you can change to version 2 when the device is in client mode if no extended VLANs exist, no private VLANs exist, and no hidden password was configured.

Caution VTP version 1 and VTP version 2 are not interoperable on devices in the same VTP domain. Do not enable VTP version 2 unless every device in the VTP domain supports version 2.

- In TrCRF and TrBRF Token Ring environments, you must enable VTP version 2 or VTP version 3 for Token Ring VLAN switching to function properly. For Token Ring and Token Ring-Net media, disable VTP version 2.

Caution In VTP version 3, both the primary and secondary servers can exist on an instance in the domain.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vtp version** {**1** | **2** | **3**} **Example:** `Device(config)# vtp version 2`	Enables the VTP version on the device. The default is VTP version 1.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show vtp status** **Example:**	Verifies that the configured VTP version is enabled.

	Command or Action	Purpose
	`Device# show vtp status`	
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Enabling VTP Pruning

Before you begin

VTP pruning is not designed to function in VTP transparent mode. If one or more devices in the network are in VTP transparent mode, you should do one of these actions:

- Turn off VTP pruning in the entire network.
- Turn off VTP pruning by making all VLANs on the trunk of the device upstream to the VTP transparent device pruning ineligible.

To configure VTP pruning on an interface, use the **switchport trunk pruning vlan** interface configuration command. VTP pruning operates when an interface is trunking. You can set VLAN pruning-eligibility, whether or not VTP pruning is enabled for the VTP domain, whether or not any given VLAN exists, and whether or not the interface is currently trunking.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vtp pruning** **Example:** `Device(config)# vtp pruning`	Enables pruning in the VTP administrative domain. By default, pruning is disabled. You need to enable pruning on only one device in VTP server mode.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show vtp status** **Example:** `Device# show vtp status`	Verifies your entries in the *VTP Pruning Mode* field of the display.

Configuring VTP on a Per-Port Basis

With VTP version 3, you can enable or disable VTP on a per-port basis. You can enable VTP only on ports that are in trunk mode. Incoming and outgoing VTP traffic are blocked, not forwarded.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Identifies an interface, and enters interface configuration mode.
Step 4	**vtp** **Example:** `Device(config-if)# vtp`	Enables VTP on the specified port.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config interface** *interface-id* **Example:** `Device# show running-config interface gigabitethernet 1/1`	Verifies the change to the port.
Step 7	**show vtp status** **Example:** `Device# show vtp status`	Verifies the configuration.

Adding a VTP Client to a VTP Domain

Follow these steps to verify and reset the VTP configuration revision number on a device *before* adding it to a VTP domain.

Before you begin

Before adding a VTP client to a VTP domain, always verify that its VTP configuration revision number is *lower* than the configuration revision number of the other devices in the VTP domain. Devices in a VTP domain always use the VLAN configuration of the device with the highest VTP configuration revision number. With VTP versions 1 and 2, adding a device that has a revision number higher than the revision number in the VTP domain can erase all VLAN information from the VTP server and VTP domain. With VTP version 3, the VLAN information is not erased.

You can use the **vtp mode transparent** global configuration command to disable VTP on the device and then to change its VLAN information without affecting the other devices in the VTP domain.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**show vtp status** **Example:**	Checks the VTP configuration revision number.

	Command or Action	Purpose
	`Device# show vtp status`	If the number is 0, add the device to the VTP domain. If the number is greater than 0, follow these substeps: • Write down the domain name. • Write down the configuration revision number. • Continue with the next steps to reset the device configuration revision number.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**vtp domain** *domain-name* **Example:** `Device(config)# vtp domain domain123`	Changes the domain name from the original one displayed in Step 1 to a new name.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. The VLAN information on the device is updated and the configuration revision number is reset to 0.
Step 6	**show vtp status** **Example:** `Device# show vtp status`	Verifies that the configuration revision number has been reset to 0.
Step 7	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 8	**vtp domain** *domain-name* **Example:** `Device(config)# vtp domain domain012`	Enters the original domain name on the device.

	Command or Action	Purpose
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. The VLAN information on the device is updated.
Step 10	**show vtp status** **Example:** `Device# show vtp status`	(Optional) Verifies that the domain name is the same as in Step 1 and that the configuration revision number is 0.

Monitoring VTP

This section describes commands used to display and monitor the VTP configuration.

You monitor VTP by displaying VTP configuration information: the domain name, the current VTP revision, and the number of VLANs. You can also display statistics about the advertisements sent and received by the device.

Table 32: VTP Monitoring Commands

Command	Purpose
show vtp counters	Displays counters about VTP messages that have been sent and received.
show vtp devices [**conflict**]	Displays information about all VTP version 3 devices in the domain. Conflicts are VTP version 3 devices with conflicting primary servers. The **show vtp devices** command does not display information when the device is in transparent or off mode.
show vtp interface [*interface-id*]	Displays VTP status and configuration for all interfaces or the specified interface.
show vtp password	Displays whether the VTP password is configured or not.
show vtp status	Displays the VTP device configuration information.

Configuration Examples for VTP

The following section shows a VTP configuration example:

Example: Configuring a Device as the Primary Server

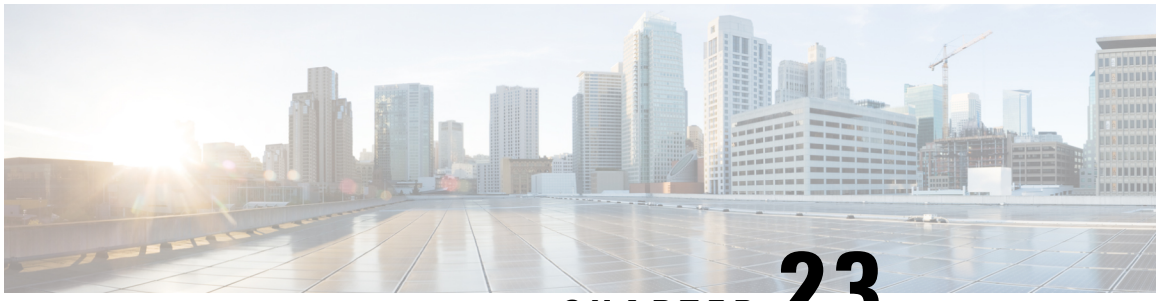
This example shows how to configure a device as the primary server for the VLAN database (the default) when a hidden or secret password was configured:

```
Device# vtp primary vlan
Enter VTP password: mypassword
This switch is becoming Primary server for vlan feature in the VTP  domain

VTP Database Conf Switch ID      Primary Server Revision System Name
------------ ---- --------------- -------------- -------- --------------------
VLANDB       Yes  00d0.00b8.1400=00d0.00b8.1400 1        stp7

Do you want to continue (y/n) [n]? y
```

Where to Go Next

After configuring VTP, you can configure the following:

- VLANs
- VLAN trunking
- Voice VLANs
- Private VLANs



CHAPTER 23

Configuring VLANs

Prerequisites for VLANs

The following are prerequisites and considerations for configuring VLANs:

- Before you create VLANs, you must decide whether to use VLAN Trunking Protocol (VTP) to maintain global VLAN configuration for your network.
- A VLAN should be present in the device to be able to add it to the VLAN group.

Restrictions for VLANs

The following are restrictions for VLANs:

- The number of Spanning Tree Protocol (STP) virtual ports in the per-VLAN spanning-tree (PVST) or rapid PVST mode is based on the number of trunks, multiplied by the number of active VLANs, plus the number of access ports.

  STP virtual ports = trunks * active VLANs on trunk + number of non-trunk ports.

  Consider the following examples:

  - If a switch has 40 trunk ports (100 active VLANs on each trunk) and 8 access ports, the number of STP virtual ports on this switch would be: 40 * 100 + 8 = 4,008.
  - If a switch has 8 trunk ports (200 active VLANs on each trunk) and 40 access ports, the number of STP virtual ports on this switch would be: 8 * 200 + 40 = 1,640

- The device supports IEEE 802.1Q trunking methods for sending VLAN traffic over Ethernet ports.
- The interface VLAN already has an MAC address assigned by default. You can override the interface VLAN MAC address by using the **mac-address** command. If this command is configured on a single

SVI or router port that requires Layer 3 injected packets, all other SVIs or routed ports on the device also must be configured with the same first four most significant bits (4MSB) of the MAC address. For example, if you set the MAC address of any SVI to xxxx.yyyy.zzzz, set the MAC address of all other SVIs to start with xxxx.yyyy. If Layer 3 injected packets are not used, this restriction does not apply.

- Once a range of interfaces has been bundled, any VLAN interface configuration change must be done only on a port channel. Otherwise, the interfaces will get suspended.

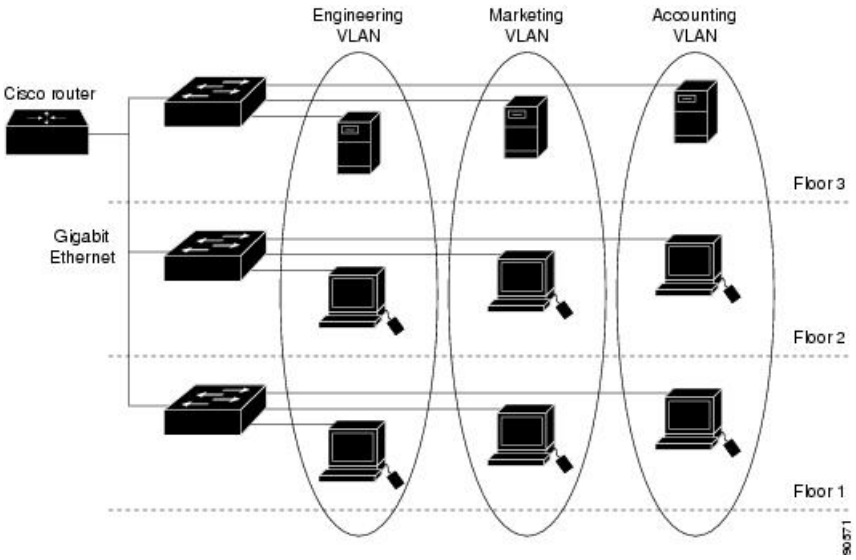
Information About VLANs

The following sections provides information about VLANs:

Logical Networks

A VLAN is a switched network that is logically segmented by function, project team, or application, without regard to the physical locations of the users. VLANs have the same attributes as physical LANs, but you can group end stations even if they are not physically located on the same LAN segment. Any device port can belong to a VLAN, and unicast, broadcast, and multicast packets are forwarded and flooded only to end stations in the VLAN. Each VLAN is considered a logical network, and packets destined for stations that do not belong to the VLAN must be forwarded through a router or a device supporting fallback bridging. Because a VLAN is considered a separate logical network, it contains its own bridge Management Information Base (MIB) information and can support its own implementation of spanning tree.

Figure 33: VLANs as Logically Defined Networks



VLANs are often associated with IP subnetworks. For example, all the end stations in a particular IP subnet belong to the same VLAN. Interface VLAN membership on the device is assigned manually on an interface-by-interface basis. When you assign device interfaces to VLANs by using this method, it is known as interface-based, or static, VLAN membership.

Traffic between VLANs must be routed.

The device can route traffic between VLANs by using device virtual interfaces (SVIs). An SVI must be explicitly configured and assigned an IP address to route traffic between VLANs.

Supported VLANs

The device supports VLANs in VTP client, server, and transparent modes. VLANs are identified by a number from 1 to 4094. VLAN 1 is the default VLAN and is created during system initialization.

- You can configure up to 1024 VLANs in a loop-free topology.
- You can configure up to 128 VLANs, when STP is enabled.
- You can configure up to 512 vlans in ring or loop based topology, when STP is disabled.

Note VLAN IDs 1002 through 1005 are reserved for Token Ring and FDDI VLANs. All of the VLANs except 1002 to 1005 are available for user configuration.

VLAN Port Membership Modes

You configure a port to belong to a VLAN by assigning a membership mode that specifies the kind of traffic the port carries and the number of VLANs to which it can belong.

When a port belongs to a VLAN, the device learns and manages the addresses associated with the port on a per-VLAN basis.

Table 33: Port Membership Modes and Characteristics

Membership Mode	VLAN Membership Characteristics	VTP Characteristics
Static-access	A static-access port can belong to one VLAN and is manually assigned to that VLAN.	VTP is not required. If you do not want VTP to globally propagate information, set the VTP mode to transparent. To participate in VTP, there must be at least one trunk port on the device connected to a trunk port of a second device.
Trunk (IEEE 802.1Q) : • IEEE 802.1Q—Industry-standard trunking encapsulation.	A trunk port is a member of all VLANs by default, including extended-range VLANs, but membership can be limited by configuring the allowed-VLAN list. You can also modify the pruning-eligible list to block flooded traffic to VLANs on trunk ports that are included in the list.	VTP is recommended but not required. VTP maintains VLAN configuration consistency by managing the addition, deletion, and renaming of VLANs on a network-wide basis. VTP exchanges VLAN configuration messages with other devices over trunk links.

Membership Mode	VLAN Membership Characteristics	VTP Characteristics
Voice VLAN	A voice VLAN port is an access port attached to a Cisco IP Phone, configured to use one VLAN for voice traffic and another VLAN for data traffic from a device attached to the phone.	VTP is not required; it has no effect on a voice VLAN.

VLAN Configuration Files

Configurations for VLAN IDs 1 to 1005 are written to the vlan.dat file (VLAN database), and you can display them by entering the **show vlan** privileged EXEC command. The vlan.dat file is stored in flash memory. If the VTP mode is transparent, they are also saved in the device running configuration file.

You use the interface configuration mode to define the port membership mode and to add and remove ports from VLANs. The results of these commands are written to the running-configuration file, and you can display the file by entering the **show running-config** privileged EXEC command.

When you save VLAN and VTP information (including extended-range VLAN configuration information) in the startup configuration file and reboot the device, the device configuration is selected as follows:

- If the VTP mode is transparent in the startup configuration, and the VLAN database and the VTP domain name from the VLAN database matches that in the startup configuration file, the VLAN database is ignored (cleared), and the VTP and VLAN configurations in the startup configuration file are used. The VLAN database revision number remains unchanged in the VLAN database.
- If the VTP mode or domain name in the startup configuration does not match the VLAN database, the domain name and VTP mode and configuration for the VLAN IDs 1 to 1005 use the VLAN database information.
- In VTP versions 1 and 2, if VTP mode is server, the domain name and VLAN configuration for VLAN IDs 1 to 1005 use the VLAN database information. VTP version 3 also supports VLANs 1006 to 4094.

Note Ensure that you delete the vlan.dat file along with the configuration files before you reset the switch configuration using **write erase** command. This ensures that the switch reboots correctly on a reset.

Normal-Range VLAN Configuration Guidelines

Normal-range VLANs are VLANs with IDs from 1 to 1005.

Follow these guidelines when creating and modifying normal-range VLANs in your network:

- Normal-range VLANs are identified with a number between 1 and 1001. VLAN numbers 1002 through 1005 are reserved for Token Ring and FDDI VLANs.
- VLAN configurations for VLANs 1 to 1005 are always saved in the VLAN database. If the VTP mode is transparent, VTP and VLAN configurations are also saved in the device running configuration file.

- If the device is in VTP server or VTP transparent mode, you can add, modify or remove configurations for VLANs 2 to 1001 in the VLAN database. (VLAN IDs 1 and 1002 to 1005 are automatically created and cannot be removed.)
- Before you can create a VLAN, the device must be in VTP server mode or VTP transparent mode. If the device is a VTP server, you must define a VTP domain or VTP will not function.
- The device does not support Token Ring or FDDI media. The device does not forward FDDI, FDDI-Net, TrCRF, or TrBRF traffic, but it does propagate the VLAN configuration through VTP.
- A fixed number of spanning tree instances are supported on the device (See the DataSheet for the latest information).

  If you have already used all available spanning-tree instances on a device, adding another VLAN anywhere in the VTP domain creates a VLAN on that device that is not running spanning-tree. If you have the default allowed list on the trunk ports of that device (which is to allow all VLANs), the new VLAN is carried on all trunk ports. Depending on the topology of the network, this could create a loop in the new VLAN that would not be broken, particularly if there are several adjacent devices that all have run out of spanning-tree instances. You can prevent this possibility by setting allowed lists on the trunk ports of devices that have used up their allocation of spanning-tree instances.

  If the number of VLANs on the device exceeds the number of supported spanning-tree instances, we recommend that you configure the IEEE 802.1s Multiple STP (MSTP) on your device to map multiple VLANs to a single spanning-tree instance.

Extended-Range VLAN Configuration Guidelines

Extended-range VLANs are VLANs with IDs from 1006 to 4094.

Follow these guidelines when creating extended-range VLANs:

- VLAN IDs in the extended range are not saved in the VLAN database and are not recognized by VTP unless the device is running VTP version 3.
- You cannot include extended-range VLANs in the pruning eligible range.
- For VTP version 1 or 2, you can set the VTP mode to transparent in global configuration mode. You should save this configuration to the startup configuration so that the device boots up in VTP transparent mode. Otherwise, you lose the extended-range VLAN configuration if the device resets. If you create extended-range VLANs in VTP version 3, you cannot convert to VTP version 1 or 2.

How to Configure VLANs

The following sections provide information about configuring Normal-Range VLANs and Extended-Range VLANs:

How to Configure Normal-Range VLANs

You can set these parameters when you create a new normal-range VLAN or modify an existing VLAN in the VLAN database:

- VLAN ID

- VLAN name
- VLAN type
  - Ethernet
  - Fiber Distributed Data Interface [FDDI]
  - FDDI network entity title [NET]
  - TrBRF or TrCRF
  - Token Ring
  - Token Ring-Net
- VLAN state (active or suspended)
- Security Association Identifier (SAID)
- Bridge identification number for TrBRF VLANs
- Ring number for FDDI and TrCRF VLANs
- Parent VLAN number for TrCRF VLANs
- Spanning Tree Protocol (STP) type for TrCRF VLANs
- VLAN number to use when translating from one VLAN type to another

You can cause inconsistency in the VLAN database if you attempt to manually delete the vlan.dat file. If you want to modify the VLAN configuration, follow the procedures in this section.

Creating or Modifying an Ethernet VLAN

Before you begin

With VTP version 1 and 2, if the device is in VTP transparent mode, you can assign VLAN IDs greater than 1006, but they are not added to the VLAN database.

The device supports only Ethernet interfaces. Because FDDI and Token Ring VLANs are not locally supported, you only configure FDDI and Token Ring media-specific characteristics for VTP global advertisements to other devices.

Although the device does not support Token Ring connections, a remote device with Token Ring connections could be managed from one of the supported devices. Devices running VTP Version 2 advertise information about these Token Ring VLANs:

- Token Ring TrBRF VLANs
- Token Ring TrCRF VLANs

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**vlan** *vlan-id* **Example:** `Device(config)# vlan 20`	Enters a VLAN ID, and enters VLAN configuration mode. Enter a new VLAN ID to create a VLAN, or enter an existing VLAN ID to modify that VLAN. **Note** The available VLAN ID range for this command is 1 to 4094.
Step 3	**name** *vlan-name* **Example:** `Device(config-vlan)# name test20`	(Optional) Enters a name for the VLAN. If no name is entered for the VLAN, the default is to append the *vlan-id* value with leading zeros to the word VLAN. For example, VLAN0004 is a default VLAN name for VLAN 4.
Step 4	**media** { **ethernet** | **fd-net** | **fddi** | **tokenring** | **trn-net** } **Example:** `Device(config-vlan)# media ethernet`	Configures the VLAN media type. Command options include: • **ethernet**—Sets the VLAN media type as Ethernet. • **fd-net**—Sets the VLAN media type as FDDI net. • **fddi**—Sets the VLAN media type as FDDI. • **tokenring**—Sets the VLAN media type as Token Ring. • **trn-net**—Sets the VLAN media type as Token Ring net.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
Step 7	**show vlan** {**name** *vlan-name* | **id** *vlan-id*} **Example:** `Device# show vlan name test20` `or` `Device# show vlan id 20`	Verifies your entries.

Deleting a VLAN

When you delete a VLAN from a device that is in VTP server mode, the VLAN is removed from the VLAN database for all devices in the VTP domain. When you delete a VLAN from a device that is in VTP transparent mode, the VLAN is deleted only on that specific device .

You cannot delete the default VLANs for the different media types: Ethernet VLAN 1 and FDDI or Token Ring VLANs 1002 to 1005.

Caution When you delete a VLAN, any ports assigned to that VLAN become inactive. They remain associated with the VLAN (and thus inactive) until you assign them to a new VLAN.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no vlan** *vlan-id* **Example:** `Device(config)# no vlan 4`	Removes the VLAN by entering the VLAN ID.
Step 4	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
Step 5	**show vlan brief** **Example:** `Device# show vlan brief`	Verifies the VLAN removal.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Assigning Static-Access Ports to a VLAN

You can assign a static-access port to a VLAN without having VTP globally propagate VLAN configuration information by disabling VTP (VTP transparent mode).

If you assign an interface to a VLAN that does not exist, the new VLAN is created. In case of interface templates, make sure to create the VLAN explicitly before applying the command via templates.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Enters the interface to be added to the VLAN.
Step 4	**switchport mode access** **Example:**	Defines the VLAN membership mode for the port (Layer 2 access port).

	Command or Action	Purpose
	`Device(config-if)# switchport mode access`	
Step 5	**switchport access vlan** *vlan-id* **Example:** `Device(config-if)# switchport access vlan 2`	Assigns the port to a VLAN. Valid VLAN IDs are 1 to 4094.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config interface** *interface-id* **Example:** `Device# show running-config interface gigabitethernet1/1`	Verifies the VLAN membership mode of the interface.
Step 8	**show interfaces** *interface-id* **switchport** **Example:** `Device# show interfaces gigabitethernet1/1 switchport`	Verifies your entries in the *Administrative Mode* and the *Access Mode VLAN* fields of the display.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

How to Configure Extended-Range VLANs

Extended-range VLANs enable service providers to extend their infrastructure to a greater number of customers. The extended-range VLAN IDs are allowed for any **switchport** commands that allow VLAN IDs.

With VTP version 1 or 2, extended-range VLAN configurations are not stored in the VLAN database, but because VTP mode is transparent, they are stored in the device running configuration file, and you can save the configuration in the startup configuration file. Extended-range VLANs created in VTP version 3 are stored in the VLAN database.

Creating an Extended-Range VLAN

Procedure

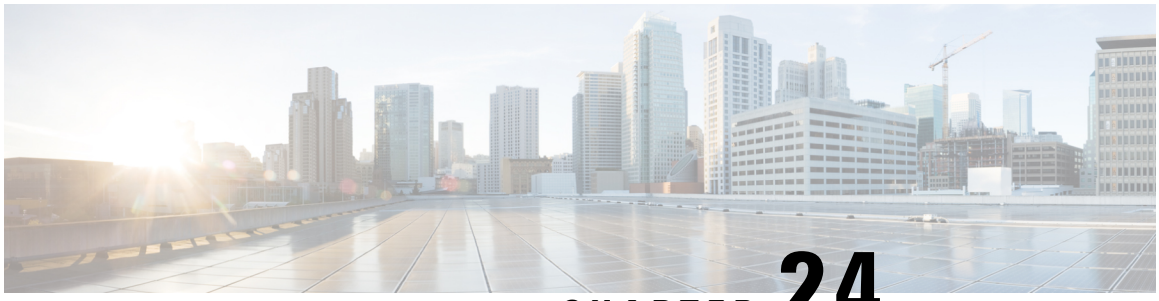
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan** *vlan-id* **Example:** `Device(config)# vlan 2000` `Device(config-vlan)#`	Enters an extended-range VLAN ID and enters VLAN configuration mode. The range is 1006 to 4094.
Step 4	**remote-span** **Example:** `Device(config-vlan)# remote-span`	(Optional) Configures the VLAN as the RSPAN VLAN.
Step 5	**exit** **Example:** `Device(config-vlan)# exit` `Device(config)#`	Returns to configuration mode.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show vlan id** *vlan-id* **Example:** `Device# show vlan id 2000`	Verifies that the VLAN has been created.

	Command or Action	Purpose
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring VLANs

Table 34: Privileged EXEC show Commands

Command	Purpose
show interfaces [**vlan** *vlan-id*]	Displays characteristics for all interfaces or for the specified VLAN configured on the device .
show vlan [**access-map** *name* | **brief** | **dot1q** { **tag native** } | **filter** [**access-map** | **vlan**] | **group** [**group-name** *name*] | **id** *vlan-id* | **ifindex** | **mtu** | **name** *name* | **private-vlan remote-span** | **summary**]	Displays parameters for all VLANs or the specified VLAN on the device. The following command options are available: • **access-map**—Displays the VLAN access-maps. • **brief**—Displays VTP VLAN status in brief. • **dot1q**—Displays the dot1q parameters. • **filter**—Displays VLAN filter information. • **group**—Displays the VLAN group with its name and the connected VLANs that are available. • **id**—Displays VTP VLAN status by identification number. • **ifindex**—Displays SNMP ifIndex. • **mtu**—Displays VLAN MTU information. • **name**—Displays the VTP VLAN information by specified name. • **private-vlan**—Displays private VLAN information. • **remote-span**–Displays the remote SPAN VLANs. • **summary**—Displays a summary of VLAN information.



CHAPTER 24

Configuring Voice VLANs

The following sections provide information about configuring Voice VLANs:

Prerequisites for Voice VLANs

The following are the prerequisites for voice VLANs:

- Voice VLAN configuration is only supported on device access ports; voice VLAN configuration is not supported on trunk ports.

Note Trunk ports can carry any number of voice VLANs, similar to regular VLANs. The configuration of voice VLANs is not supported on trunk ports.

- Before you enable voice VLAN, enable QoS on the device by entering the **trust device cisco-phone** interface configuration command. If you use the auto QoS feature, these settings are automatically configured.
- You must enable CDP on the device port connected to the Cisco IP Phone to send the configuration to the phone. (CDP is globally enabled by default on all device interfaces.)

Restrictions for Voice VLANs

You cannot configure static secure MAC addresses in the voice VLAN.

Information About Voice VLAN

The following sections provide information about Voice VLAN:

Voice VLANs

The voice VLAN feature enables access ports to carry IP voice traffic from an IP phone. When the device is connected to a Cisco 7960 IP Phone, the phone sends voice traffic with Layer 3 IP precedence and Layer 2 class of service (CoS) values, which are both set to 5 by default. Because the sound quality of an IP phone call can deteriorate if the data is unevenly sent, the device supports quality of service (QoS) based on IEEE 802.1p CoS. QoS uses classification and scheduling to send network traffic from the device in a predictable manner.

The Cisco 7960 IP Phone is a configurable device, and you can configure it to forward traffic with an IEEE 802.1p priority. You can configure the device to trust or override the traffic priority assigned by a Cisco IP Phone.

Cisco IP Phone Voice Traffic

You can configure an access port with an attached Cisco IP Phone to use one VLAN for voice traffic and another VLAN for data traffic from a device attached to the phone. You can configure access ports on the device to send Cisco Discovery Protocol (CDP) packets that instruct an attached phone to send voice traffic to the device in any of these ways:

- In the voice VLAN tagged with a Layer 2 CoS priority value
- In the access VLAN tagged with a Layer 2 CoS priority value
- In the access VLAN, untagged (no Layer 2 CoS priority value)

Note In all configurations, the voice traffic carries a Layer 3 IP precedence value (the default is 5 for voice traffic and 3 for voice control traffic).

Cisco IP Phone Data Traffic

The device can also process tagged data traffic (traffic in IEEE 802.1Q or IEEE 802.1p frame types) from the device attached to the access port on the Cisco IP Phone. You can configure Layer 2 access ports on the device to send CDP packets that instruct the attached phone to configure the phone access port in one of these modes:

- In trusted mode, all traffic received through the access port on the Cisco IP Phone passes through the phone unchanged.
- In untrusted mode, all traffic in IEEE 802.1Q or IEEE 802.1p frames received through the access port on the Cisco IP Phone receive a configured Layer 2 CoS value. The default Layer 2 CoS value is 0. Untrusted mode is the default.

Note Untagged traffic from the device attached to the Cisco IP Phone passes through the phone unchanged, regardless of the trust state of the access port on the phone.

Voice VLAN Configuration Guidelines

- Because a Cisco 7960 IP Phone also supports a connection to a PC or other device, a port connecting the device to a Cisco IP Phone can carry mixed traffic. You can configure a port to decide how the Cisco IP Phone carries voice traffic and data traffic.
- The voice VLAN should be present and active on the device for the IP phone to correctly communicate on the voice VLAN. Use the **show vlan** privileged EXEC command to see if the VLAN is present (listed in the display). If the VLAN is not listed, create the voice VLAN.
- The Port Fast feature is automatically enabled when voice VLAN is configured. When you disable voice VLAN, the Port Fast feature is not automatically disabled.
- If the Cisco IP Phone and a device attached to the phone are in the same VLAN, they must be in the same IP subnet. These conditions indicate that they are in the same VLAN:
  - They both use IEEE 802.1p or untagged frames.
  - The Cisco IP Phone uses IEEE 802.1p frames, and the device uses untagged frames.
  - The Cisco IP Phone uses untagged frames, and the device uses IEEE 802.1p frames.
  - The Cisco IP Phone uses IEEE 802.1Q frames, and the voice VLAN is the same as the access VLAN.
- The Cisco IP Phone and a device attached to the phone cannot communicate if they are in the same VLAN and subnet but use different frame types because traffic in the same subnet is not routed (routing would eliminate the frame type difference).
- Voice VLAN ports can also be these port types:
  - Dynamic access port.
  - IEEE 802.1x authenticated port.

Note If you enable IEEE 802.1x on an access port on which a voice VLAN is configured and to which a Cisco IP Phone is connected, the phone loses connectivity to the device for up to 30 seconds.

  - Protected port.
  - A source or destination port for a SPAN or RSPAN session.
  - Secure port.

Note When you enable port security on an interface that is also configured with a voice VLAN, you must set the maximum allowed secure addresses on the port to two plus the maximum number of secure addresses allowed on the access VLAN. When the port is connected to a Cisco IP Phone, the phone requires up to two MAC addresses. The phone address is learned on the voice VLAN and might also be learned on the access VLAN. Connecting a PC to the phone requires additional MAC addresses.

How to Configure Voice VLANs

The following sections provide information about configuring Voice VLANs:

Configuring Cisco IP Phone Voice Traffic

You can configure a port connected to the Cisco IP Phone to send CDP packets to the phone to configure the way in which the phone sends voice traffic. The phone can carry voice traffic in IEEE 802.1Q frames for a specified voice VLAN with a Layer 2 CoS value. It can use IEEE 802.1p priority tagging to give voice traffic a higher priority and forward all voice traffic through the native (access) VLAN. The Cisco IP Phone can also send untagged voice traffic or use its own configuration to send voice traffic in the access VLAN. In all configurations, the voice traffic carries a Layer 3 IP precedence value (the default is 5).

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface connected to the phone, and enters interface configuration mode.
Step 3	**trust device cisco-phone** **Example:** `Device(config-if)# trust device cisco-phone`	Configures the interface to trust incoming traffic packets for the Cisco IP phone.
Step 4	**switchport voice vlan** {*vlan-id* | **dot1p** | **none** | **untagged**} **Example:** `Device(config-if)# switchport voice vlan dot1p`	Configures the voice VLAN. • *vlan-id*—Configures the phone to forward all voice traffic through the specified VLAN. By default, the Cisco IP Phone forwards the voice traffic with an IEEE 802.1Q priority of 5. Valid VLAN IDs are 1 to 4094. • **dot1p**—Configures the device to accept voice and data IEEE 802.1p priority frames tagged with VLAN ID 0 (the native VLAN). By default, the device drops all voice and data traffic tagged with VLAN

	Command or Action	Purpose
		0. If configured for 802.1p the Cisco IP Phone forwards the traffic with an IEEE 802.1p priority of 5. • **none**—Allows the phone to use its own configuration to send untagged voice traffic. • **untagged**—Configures the phone to send untagged voice traffic.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	Use one of the following: • **show interfaces** *interface-id* **switchport** • **show running-config interface** *interface-id* **Example:** `Device# show interfaces gigabitethernet1/1 switchport` or `Device# show running-config interface gigabitethernet1/1`	Verifies your voice VLAN entries or your QoS and voice VLAN entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring the Priority of Incoming Data Frames

You can connect a PC or other data device to a Cisco IP Phone port. To process tagged data traffic (in IEEE 802.1Q or IEEE 802.1p frames), you can configure the device to send CDP packets to instruct the phone how to send data packets from the device attached to the access port on the Cisco IP Phone. The PC can generate packets with an assigned CoS value. You can configure the phone to not change (trust) or to override (not trust) the priority of frames arriving on the phone port from connected devices.

Follow these steps to set the priority of data traffic received from the non-voice port on the Cisco IP Phone:

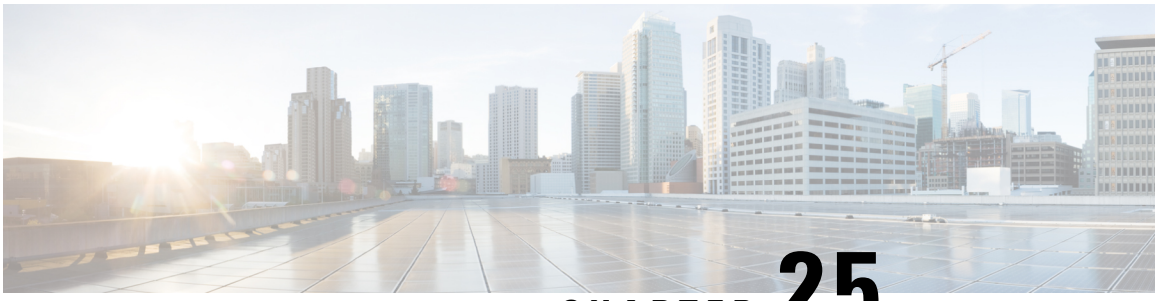
Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface connected to the Cisco IP Phone, and enters interface configuration mode.
Step 4	**switchport priority extend** {**cos** *value* | **trust**} **Example:** `Device(config-if)# switchport priority extend trust`	Sets the priority of data traffic received from the Cisco IP Phone access port: • **cos** *value*—Configures the phone to override the priority received from the PC or the attached device with the specified CoS value. The value is a number from 0 to 7, with 7 as the highest priority. The default priority is **cos** 0. • **trust**—Configures the phone access port to trust the priority received from the PC or the attached device.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show interfaces** *interface-id* **switchport** **Example:** `Device# show interfaces gigabitethernet1/1 switchport`	Verifies your entries.

	Command or Action	Purpose
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring Voice VLAN

To display voice VLAN configuration for an interface, use the **show interfaces** *interface-id* **switchport** privileged EXEC command.



CHAPTER 25

Configuring VLAN Trunks

Information About VLAN Trunks

The following sections provide information about VLAN Trunks:

Trunking Overview

A trunk is a point-to-point link between one or more Ethernet interfaces and another networking device such as a router or a controller. Ethernet trunks carry the traffic of multiple VLANs over a single link, and you can extend the VLANs across an entire network.

IEEE 802.1Q— Industry-standard trunking encapsulation is available on all Ethernet interfaces.

Trunking Modes

Ethernet trunk interfaces support different trunking modes. You can set an interface as trunking or nontrunking or to negotiate trunking with the neighboring interface. To autonegotiate trunking, the interfaces must be in the same VTP domain.

Trunk negotiation is managed by the Dynamic Trunking Protocol (DTP). However, some internetworking devices might forward DTP frames improperly, which could cause misconfigurations.

Layer 2 Interface Modes

Table 35: Layer 2 Interface Modes

Mode	Function
switchport mode access	Puts the interface (access port) into permanent nontrunking mode and negotiates to convert the link into a nontrunk link. The interface becomes a nontrunk interface regardless of whether or not the neighboring interface is a trunk interface.
switchport mode dynamic auto	Makes the interface able to convert the link to a trunk link. The interface becomes a trunk interface if the neighboring interface is set to **trunk** or **desirable** mode. The default switchport mode for all Ethernet interfaces is **dynamic auto**.
switchport mode dynamic desirable	Makes the interface actively attempt to convert the link to a trunk link. The interface becomes a trunk interface if the neighboring interface is set to **trunk**, **desirable**, or **auto** mode.
switchport mode trunk	Puts the interface into permanent trunking mode and negotiates to convert the neighboring link into a trunk link. The interface becomes a trunk interface even if the neighboring interface is not a trunk interface.
switchport nonegotiate	Prevents the interface from generating DTP frames. You can use this command only when the interface switchport mode is **access** or **trunk**. You must manually configure the neighboring interface as a trunk interface to establish a trunk link.
switchport mode private-vlan	Configures the private VLAN mode.

Allowed VLANs on a Trunk

By default, a trunk port sends traffic to and receives traffic from all VLANs. All VLAN IDs, 1 to 4094, are allowed on each trunk. However, you can remove VLANs from the allowed list, preventing traffic from those VLANs from passing over the trunk.

To reduce the risk of spanning-tree loops or storms, you can disable VLAN 1 on any individual VLAN trunk port by removing VLAN 1 from the allowed list. When you remove VLAN 1 from a trunk port, the interface continues to send and receive management traffic, for example, Cisco Discovery Protocol (CDP), Port Aggregation Protocol (PAgP), Link Aggregation Control Protocol (LACP), DTP, and VTP in VLAN 1.

If a trunk port with VLAN 1 disabled is converted to a nontrunk port, it is added to the access VLAN. If the access VLAN is set to 1, the port will be added to VLAN 1, regardless of the **switchport trunk allowed** setting. The same is true for any VLAN that has been disabled on the port.

A trunk port can become a member of a VLAN if the VLAN is enabled, if VTP knows of the VLAN, and if the VLAN is in the allowed list for the port. When VTP detects a newly enabled VLAN and the VLAN is in the allowed list for a trunk port, the trunk port automatically becomes a member of the enabled VLAN. When VTP detects a new VLAN and the VLAN is not in the allowed list for a trunk port, the trunk port does not become a member of the new VLAN.

Load Sharing on Trunk Ports

Load sharing divides the bandwidth supplied by parallel trunks connecting devices. To avoid loops, STP normally blocks all but one parallel link between the devices. Using load sharing, you divide the traffic between the links according to which VLAN the traffic belongs.

You configure load sharing on trunk ports by using STP port priorities or STP path costs. For load sharing using STP port priorities, both load-sharing links must be connected to the same device. For load sharing using STP path costs, each load-sharing link can be connected to the same device or to two different devices.

Network Load Sharing Using STP Priorities

When two ports on the same device form a loop, the device uses the STP port priority to decide which port is enabled and which port is in a blocking state. You can set the priorities on a parallel trunk port so that the port carries all the traffic for a given VLAN. The trunk port with the higher priority (lower values) for a VLAN is forwarding traffic for that VLAN. The trunk port with the lower priority (higher values) for the same VLAN remains in a blocking state for that VLAN. One trunk port sends or receives all traffic for the VLAN.

Network Load Sharing Using STP Path Cost

You can configure parallel trunks to share VLAN traffic by setting different path costs on a trunk and associating the path costs with different sets of VLANs, blocking different ports for different VLANs. The VLANs keep the traffic separate and maintain redundancy in the event of a lost link.

Feature Interactions

Trunking interacts with other features in these ways:

- A trunk port cannot be a secure port.
- Trunk ports can be grouped into EtherChannel port groups, but all trunks in the group must have the same configuration. When a group is first created, all ports follow the parameters set for the first port to be added to the group. If you change the configuration of one of these parameters, the device propagates the setting that you entered to all ports in the group:
  - Allowed-VLAN list.
  - STP port priority for each VLAN.
  - STP Port Fast setting.
  - Trunk status:

    If one port in a port group ceases to be a trunk, all ports cease to be trunks.
- If you try to enable IEEE 802.1x on a trunk port, an error message appears, and IEEE 802.1x is not enabled. If you try to change the mode of an IEEE 802.1x-enabled port to trunk, the port mode is not changed.
- A port in dynamic mode can negotiate with its neighbor to become a trunk port. If you try to enable IEEE 802.1x on a dynamic port, an error message appears, and IEEE 802.1x is not enabled. If you try to change the mode of an IEEE 802.1x-enabled port to dynamic, the port mode is not changed.

Prerequisites for VLAN Trunks

The IEEE 802.1Q trunks impose these limitations on the trunking strategy for a network:

- In a network of Cisco devices connected through IEEE 802.1Q trunks, the devices maintain one spanning-tree instance for each VLAN allowed on the trunks. Non-Cisco devices might support one spanning-tree instance for all VLANs.

  When you connect a Cisco device to a non-Cisco device through an IEEE 802.1Q trunk, the Cisco device combines the spanning-tree instance of the VLAN of the trunk with the spanning-tree instance of the non-Cisco IEEE 802.1Q device. However, spanning-tree information for each VLAN is maintained by Cisco devices separated by a cloud of non-Cisco IEEE 802.1Q devices. The non-Cisco IEEE 802.1Q cloud separating the Cisco device is treated as a single trunk link between the devices.

- Make sure the native VLAN for an IEEE 802.1Q trunk is the same on both ends of the trunk link. If the native VLAN on one end of the trunk is different from the native VLAN on the other end, spanning-tree loops might result.

- Disabling spanning tree on the native VLAN of an IEEE 802.1Q trunk without disabling spanning tree on every VLAN in the network can potentially cause spanning-tree loops. We recommend that you leave spanning tree enabled on the native VLAN of an IEEE 802.1Q trunk or disable spanning tree on every VLAN in the network. Make sure your network is loop-free before disabling spanning tree.

Restrictions for VLAN Trunks

The following are restrictions for VLAN trunks:

- A trunk port cannot be a secure port.

- Trunk ports can be grouped into EtherChannel port groups, but all trunks in the group must have the same configuration. When a group is first created, all ports follow the parameters set for the first port to be added to the group. If you change the configuration of one of these parameters, the device propagates the setting that you entered to all ports in the group:

  - Allowed-VLAN list.
  - STP port priority for each VLAN.
  - STP Port Fast setting.
  - Trunk status:

    If one port in a port group ceases to be a trunk, all ports cease to be trunks.

- If you try to enable IEEE 802.1x on a trunk port, an error message appears, and IEEE 802.1x is not enabled. If you try to change the mode of an IEEE 802.1x-enabled port to trunk, the port mode is not changed.

- A port in dynamic mode can negotiate with its neighbor to become a trunk port. If you try to enable IEEE 802.1x on a dynamic port, an error message appears, and IEEE 802.1x is not enabled. If you try to change the mode of an IEEE 802.1x-enabled port to dynamic, the port mode is not changed.

- Dynamic Trunking Protocol (DTP) is not supported on tunnel ports.

- The device does not support Layer 3 trunks; you cannot configure subinterfaces or use the **encapsulation** keyword on Layer 3 interfaces. The device does support Layer 2 trunks and Layer 3 VLAN interfaces, which provide equivalent capabilities.
- When native VLAN and management VLAN is configured with the same VLAN ID and a new VLAN is added as trunk port, both the new VLAN and native VLAN shifts between active and suspend state for a duration of 15 seconds. This duration is the time taken for STP to resolve all inconsistencies.
- The **switchport trunk native vlan** command is not supported.

How to Configure VLAN Trunks

To avoid trunking misconfigurations, configure interfaces connected to devices that do not support DTP to not forward DTP frames, that is, to turn off DTP.

- If you do not intend to trunk across those links, use the **switchport mode access** interface configuration command to disable trunking.
- To enable trunking to a device that does not support DTP, use the **switchport mode trunk** and **switchport nonegotiate** interface configuration commands to cause the interface to become a trunk but to not generate DTP frames.

Configuring an Ethernet Interface as a Trunk Port

This section provides information about configuring an Ethernet Interface as a trunk port:

Configuring a Trunk Port

Because trunk ports send and receive VTP advertisements, to use VTP you must ensure that at least one trunk port is configured on the device and that this trunk port is connected to the trunk port of a second device. Otherwise, the device cannot receive any VTP advertisements.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured for trunking, and enters interface configuration mode.
Step 4	**switchport mode** {**dynamic** {**auto** | **desirable**} | **trunk**} **Example:** `Device(config-if)# switchport mode dynamic desirable`	Configures the interface as a Layer 2 trunk (required only if the interface is a Layer 2 access port or tunnel port or to specify the trunking mode). • **dynamic auto**—Sets the interface to a trunk link if the neighboring interface is set to trunk or desirable mode. This is the default. • **dynamic desirable**—Sets the interface to a trunk link if the neighboring interface is set to trunk, desirable, or auto mode. • **trunk**—Sets the interface in permanent trunking mode and negotiate to convert the link to a trunk link even if the neighboring interface is not a trunk interface.
Step 5	**switchport access vlan** *vlan-id* **Example:** `Device(config-if)# switchport access vlan 200`	(Optional) Specifies the default VLAN, which is used if the interface stops trunking.
Step 6	**switchport trunk native vlan** *vlan-id* **Example:** `Device(config-if)# switchport trunk native vlan 200`	Specifies the native VLAN for IEEE 802.1Q trunks.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show interfaces** *interface-id* **switchport** **Example:**	Displays the switch port configuration of the interface in the *Administrative Mode* and the *Administrative Trunking Encapsulation* fields of the display.

	Command or Action	Purpose
	`Device# show interfaces gigabitethernet 1/1 switchport`	

Defining the Allowed VLANs on a Trunk

VLAN 1 is the default VLAN on all trunk ports in all Cisco devices, and it has previously been a requirement that VLAN 1 always be enabled on every trunk link. You can use the VLAN 1 minimization feature to disable VLAN 1 on any individual VLAN trunk link so that no user traffic (including spanning-tree advertisements) is sent or received on VLAN 1.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Configures the interface as a VLAN trunk port.
Step 5	**switchport trunk allowed vlan** { *word* | **add** | **all** | **except** | **none** | **remove**} *vlan-list* **Example:** `Device(config-if)# switchport trunk allowed vlan remove 2`	(Optional) Configures the list of VLANs allowed on the trunk. The *vlan-list* parameter is either a single VLAN number from 1 to 4094 or a range of VLANs described by two VLAN numbers, the lower one first, separated by a hyphen. Do not enter any spaces between comma-separated VLAN parameters or in hyphen-specified ranges.

	Command or Action	Purpose
		All VLANs are allowed by default.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show interfaces** *interface-id* **switchport** **Example:** `Device# show interfaces gigabitethernet 1/1 switchport`	Verifies your entries in the *Trunking VLANs Enabled* field of the display.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Changing the Pruning-Eligible List

The pruning-eligible list applies only to trunk ports. Each trunk port has its own eligibility list. VTP pruning must be enabled for this procedure to take effect.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Selects the trunk port for which VLANs should be pruned, and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**switchport trunk pruning vlan** {**add** | **except** | **none** | **remove**} *vlan-list* [,*vlan* [,*vlan* [,,,]]	Configures the list of VLANs allowed to be pruned from the trunk. For explanations about using the **add**, **except**, **none**, and **remove** keywords, see the command reference for this release. Separate non-consecutive VLAN IDs with a comma and no spaces; use a hyphen to designate a range of IDs. Valid IDs are 2 to 1001. Extended-range VLANs (VLAN IDs 1006 to 4094) cannot be pruned. VLANs that are pruning-ineligible receive flooded traffic. The default list of VLANs allowed to be pruned contains VLANs 2 to 1001.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show interfaces** *interface-id* **switchport** **Example:** `Device# show interfaces gigabitethernet 1/1 switchport`	Verifies your entries in the *Pruning VLANs Enabled* field of the display.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring the Native VLAN for Untagged Traffic

A trunk port configured with IEEE 802.1Q tagging can receive both tagged and untagged traffic. By default, the device forwards untagged traffic in the native VLAN configured for the port. The native VLAN is VLAN 1 by default.

The native VLAN can be assigned any VLAN ID.

If a packet has a VLAN ID that is the same as the outgoing port native VLAN ID, the packet is sent untagged; otherwise, the device sends the packet with a tag.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Defines the interface that is configured as the IEEE 802.1Q trunk, and enters interface configuration mode.
Step 4	**switchport trunk native vlan** *vlan-id* **Example:** `Device(config-if)# switchport trunk native vlan 12`	Configures the VLAN that is sending and receiving untagged traffic on the trunk port. For *vlan-id*, the range is 1 to 4094.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show interfaces** *interface-id* **switchport** **Example:** `Device# show interfaces gigabitethernet 1/1 switchport`	Verifies your entries in the *Trunking Native Mode VLAN* field.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Trunk Ports for Load Sharing

The following sections provide information about configuring trunk ports for load sharing:

Configuring Load Sharing Using STP Port Priorities

These steps describe how to configure a network with load sharing using STP port priorities.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode on Device A.
Step 3	**vtp domain** *domain-name* **Example:** `Device(config)# vtp domain workdomain`	Configures a VTP administrative domain. The domain name can be 1 to 32 characters.
Step 4	**vtp mode server** **Example:** `Device(config)# vtp mode server`	Configures Device A as the VTP server.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show vtp status** **Example:** `Device# show vtp status`	Verifies the VTP configuration on both Device A and Device B. In the display, check the *VTP Operating Mode* and the *VTP Domain Name* fields.
Step 7	**show vlan** **Example:**	Verifies that the VLANs exist in the database on Device A.

	Command or Action	Purpose
	`Device# show vlan`	
Step 8	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 9	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Defines the interface to be configured as a trunk, and enters interface configuration mode.
Step 10	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Configures the port as a trunk port.
Step 11	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 12	**show interfaces** *interface-id* **switchport** **Example:** `Device# show interfaces gigabitethernet 1/1 switchport`	Verifies the VLAN configuration.
Step 13	Repeat the above steps on Device A for a second port in the device.	
Step 14	Repeat the above steps on Device B to configure the trunk ports that connect to the trunk ports configured on Device A.	
Step 15	**show vlan** **Example:** `Device# show vlan`	When the trunk links come up, VTP passes the VTP and VLAN information to Device B. This command verifies that Device B has learned the VLAN configuration.

	Command or Action	Purpose
Step 16	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode on Device A.
Step 17	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Defines the interface to set the STP port priority, and enters interface configuration mode.
Step 18	**spanning-tree vlan** *vlan-range* **port-priority** *priority-value* **Example:** `Device(config-if)# spanning-tree vlan 8-10 port-priority 16`	Assigns the port priority for the VLAN range specified. Enter a port priority value from 0 to 240. Port priority values increment by 16.
Step 19	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 20	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Defines the interface to set the STP port priority, and enters interface configuration mode.
Step 21	**spanning-tree vlan** *vlan-range* **port-priority** *priority-value* **Example:** `Device(config-if)# spanning-tree vlan 3-6 port-priority 16`	Assigns the port priority for the VLAN range specified. Enter a port priority value from 0 to 240. Port priority values increment by 16.
Step 22	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 23	**show running-config** **Example:**	Verifies your entries.

	Command or Action	Purpose
	`Device# show running-config`	
Step 24	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Load Sharing Using STP Path Cost

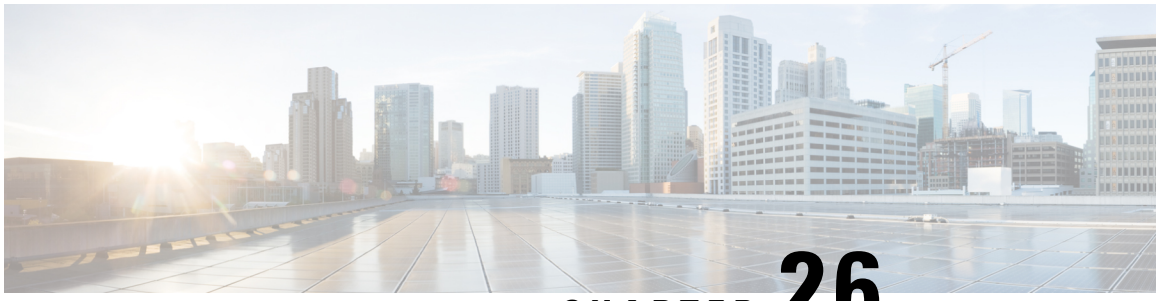
These steps describe how to configure a network with load sharing using STP path costs.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode on Device A.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Defines the interface to be configured as a trunk, and enters interface configuration mode.
Step 4	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Configures the port as a trunk port.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.

	Command or Action	Purpose
Step 6	Repeat Steps 2 through 4 on a second interface in Device A .	
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries. In the display, make sure that the interfaces are configured as trunk ports.
Step 9	**show vlan** **Example:** `Device# show vlan`	When the trunk links come up, Device A receives the VTP information from the other devices. This command verifies that Device A has learned the VLAN configuration.
Step 10	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 11	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Defines the interface on which to set the STP cost, and enters interface configuration mode.
Step 12	**spanning-tree vlan** *vlan-range* **cost** *cost-value* **Example:** `Device(config-if)# spanning-tree vlan 2-4 cost 30`	Sets the spanning-tree path cost to 30 for VLANs 2 through 4.
Step 13	**end** **Example:** `Device(config-if)# end`	Returns to global configuration mode.
Step 14	Repeat Steps 9 through 13 on the other configured trunk interface on Device A, and	

	Command or Action	Purpose
	set the spanning-tree path cost to 30 for VLANs 8, 9, and 10.	
Step 15	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 16	**show running-config** **Example:** `Device# show running-config`	Verifies your entries. In the display, verify that the path costs are set correctly for both trunk interfaces.
Step 17	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.



CHAPTER 26

Configuring Private VLANs

The following sections provide information about configuring Private VLANs:

Restrictions for Private VLANs

Note In some cases, the configuration is accepted with no error messages, but the commands have no effect.

- Do not configure fallback bridging on the device with private VLANs.
- Do not configure a remote SPAN (RSPAN) VLAN as a primary or a secondary VLAN of a private-VLAN.
- Do not configure private VLAN ports on interfaces configured for these other features:
  - Dynamic-access port VLAN membership
  - Dynamic Trunking Protocol (DTP)
  - IP Source Guard
  - IPv6 First Hop Security (FHS)
  - IPv6 Security Group (SG)
  - Multicast VLAN Registration (MVR)
  - Voice VLAN
  - Web Cache Communication Protocol (WCCP)
- Port Aggregation Protocol (PAgP) and Link Aggregation Control Protocol (LACP) are supported only for Private VLAN promiscuous trunk ports and Private VLAN isolated trunk ports.

- You can configure IEEE 802.1x port-based authentication on a private-VLAN port, but do not configure 802.1x with port security, voice VLAN, or per-user ACL on private-VLAN ports.
- A private-VLAN host or promiscuous port cannot be a SPAN destination port. If you configure a SPAN destination port as a private-VLAN port, the port becomes inactive.
- If you configure a static MAC address on a promiscuous port in the primary VLAN, you need not add the same static address to all associated secondary VLANs. Similarly, if you configure a static MAC address on a host port in a secondary VLAN, you need not add the same static MAC address to the associated primary VLAN. Also, when you delete a static MAC address from a private-VLAN port, you do not have to remove all instances of the configured MAC address from the private VLAN.

Note Dynamic MAC addresses learned in the secondary VLAN of a private VLAN are replicated to the primary VLANs. All MAC entries are learnt on secondary VLANs, even if the traffic ingresses from primary VLAN. If a MAC address is dynamically learnt in the primary VLAN, it is not replicated in the associated secondary VLANs.

- Configure Layer 3 VLAN interfaces (switch value interfaces) only for primary VLANs.
- Private VLAN configured with MACsec or Virtual Private LAN Services (VPLS) or Cisco Software-Defined Access solution on the same VLAN does not work.

Information About Private VLANs

The following sections provide information about Private VLANs:

Private VLAN Domains

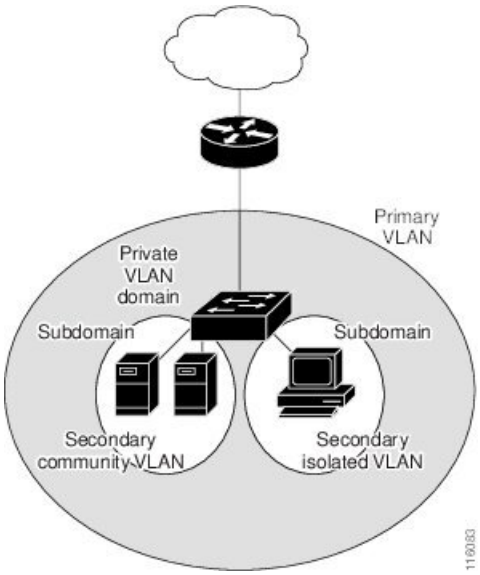
The private VLAN feature addresses two problems that service providers face when using VLANs:

- To enable IP routing, each VLAN is assigned a subnet address space or a block of addresses, which can result in wasting the unused IP addresses, and cause IP address management problems.

Figure 34: Private VLAN Domain

Using private VLANs addresses the scalability problem and provides IP address management benefits for service providers and Layer 2 security for customers. Private VLANs partition a regular VLAN domain into subdomains. A subdomain is represented by a pair of VLANs: a *primary* VLAN and a *secondary* VLAN. A private VLAN can have multiple VLAN pairs, one pair for each subdomain. All VLAN pairs in a private

VLAN share the same primary VLAN. The secondary VLAN ID differentiates one subdomain from another.



Secondary VLANs

There are two types of secondary VLANs:

- Isolated VLANs—Ports within an isolated VLAN cannot communicate with each other at the Layer 2 level.
- Community VLANs—Ports within a community VLAN can communicate with each other but cannot communicate with ports in other communities at the Layer 2 level.

Private VLANs Ports

Private VLANs provide Layer 2 isolation between ports within the same private VLAN. Private VLAN ports are access ports that are one of these types:

- Isolated—An isolated port is a host port that belongs to an isolated secondary VLAN. It has complete Layer 2 separation from other ports within the same private VLAN, except for the promiscuous ports. Private VLANs block all traffic to isolated ports except traffic from promiscuous ports. Traffic received from an isolated port is forwarded only to promiscuous ports.
- Community—A community port is a host port that belongs to a community secondary VLAN. Community ports communicate with other ports in the same community VLAN and with promiscuous ports. These interfaces are isolated at Layer 2 from all other interfaces in other communities and from isolated ports within their private VLAN.

Note Promiscuous ports are not supported.

Trunk ports carry traffic from regular VLANs and also from primary, isolated, and community VLANs.

Primary and secondary VLANs have these characteristics:

- Primary VLAN—A private VLAN has only one primary VLAN. Every port in a private VLAN is a member of the primary VLAN. The primary VLAN carries unidirectional traffic downstream from the promiscuous ports to the (isolated and community) host ports and to other promiscuous ports.
- Isolated VLAN —A private VLAN has only one isolated VLAN. An isolated VLAN is a secondary VLAN that carries unidirectional traffic upstream from the hosts toward the promiscuous ports and the gateway.
- Community VLAN—A community VLAN is a secondary VLAN that carries upstream traffic from the community ports to the promiscuous port gateways and to other host ports in the same community. You can configure multiple community VLANs in a private VLAN.

A promiscuous port can serve only one primary VLAN, one isolated VLAN, and multiple community VLANs. Layer 3 gateways are typically connected to the device through a promiscuous port. With a promiscuous port, you can connect a wide range of devices as access points to a private VLAN. For example, you can use a promiscuous port to monitor or back up all the private VLAN servers from an administration workstation.

Private VLANs in Networks

In a switched environment, you can assign an individual private VLAN and associated IP subnet to each individual or common group of end stations. The end stations need to communicate only with a default gateway to communicate outside the private VLAN.

You can use private VLANs to control access to end stations in these ways:

- Configure selected interfaces connected to end stations as isolated ports to prevent any communication at Layer 2. For example, if the end stations are servers, this configuration prevents Layer 2 communication between the servers.
- Configure interfaces connected to default gateways and selected end stations (for example, backup servers) as promiscuous ports to allow all end stations access to a default gateway.

You can extend private VLANs across multiple devices by trunking the primary, isolated, and community VLANs to other devices that support private VLANs. To maintain the security of your private VLAN configuration and to avoid other use of the VLANs configured as private VLANs, configure private VLANs on all intermediate devices, including devices that have no private VLAN ports.

IP Addressing Scheme with Private VLANs

Assigning a separate VLAN to each customer creates an inefficient IP addressing scheme:

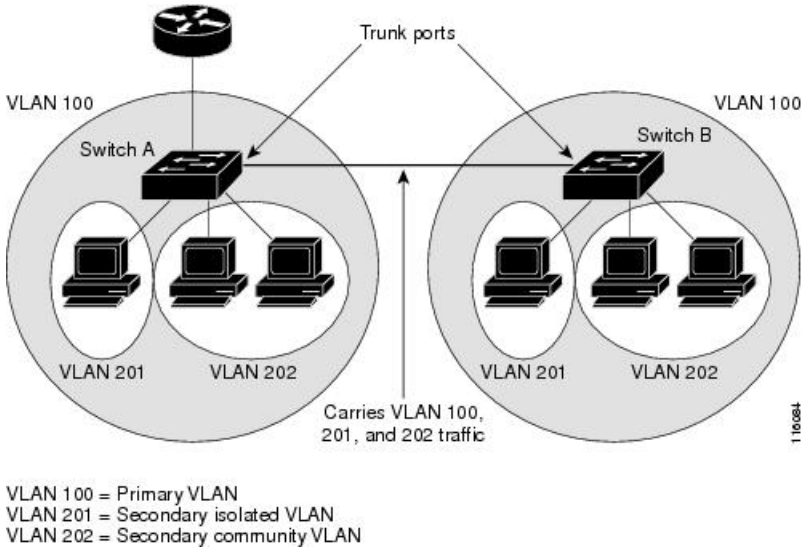
- Assigning a block of addresses to a customer VLAN can result in unused IP addresses.
- If the number of devices in the VLAN increases, the number of assigned address might not be large enough to accommodate them.

These problems are reduced by using private VLANs, where all members in the private VLAN share a common address space, which is allocated to the primary VLAN. Hosts are connected to secondary VLANs and the DHCP server assigns them IP addresses from the block of addresses allocated to the primary VLAN. Subsequent IP addresses can be assigned to customer devices in different secondary VLANs, but in the same primary VLAN. When new devices are added, the DHCP server assigns them the next available address from a large pool of subnet addresses.

Private VLANs Across Multiple Devices

Figure 35: Private VLANs Across Switches

As with regular VLANs, private VLANs can span multiple switches. A trunk port carries the primary VLAN and secondary VLANs to a neighboring switch. The trunk port treats the private VLAN as any other VLAN. A feature of private VLANs across multiple switches is that traffic from an isolated port in the Switch A does not reach an isolated port on Switch B.



Private VLANs are supported in transparent mode for VTP 1, 2 and 3. Private VLAN is also supported on server mode for VTP 3. If you have a server client setup using VTP 3, private VLANs configured on the server should be reflected on the client.

Private-VLAN Interaction with Other Features

The following sections provide information about Private-VLAN interaction with other features:

Private VLANs and Unicast, Broadcast, and Multicast Traffic

In regular VLANs, devices in the same VLAN can communicate with each other at the Layer 2 level, but devices connected to interfaces in different VLANs must communicate at the Layer 3 level. In private VLANs, the promiscuous ports are members of the primary VLAN, while the host ports belong to secondary VLANs. Because the secondary VLAN is associated with the primary VLAN, members of the these VLANs can communicate with each other at the Layer 2 level.

In a regular VLAN, broadcasts are forwarded to all ports in that VLAN. Private VLAN broadcast forwarding depends on the port sending the broadcast:

- An isolated port sends a broadcast only to the promiscuous ports or trunk ports.
- A community port sends a broadcast to all promiscuous ports, trunk ports, and ports in the same community VLAN.
- A promiscuous port sends a broadcast to all ports in the private VLAN (other promiscuous ports, trunk ports, isolated ports, and community ports).

Multicast traffic is routed or bridged across private VLAN boundaries and within a single community VLAN. Multicast traffic is not forwarded between ports in the same isolated VLAN or between ports in different secondary VLANs.

Private VLAN multicast forwarding supports the following:

- Sender can be outside the VLAN and the Receivers can be inside the VLAN domain.
- Sender can be inside the VLAN and the Receivers can be outside the VLAN domain.
- Sender and Receiver can both be in the same community VLAN.

Private VLANs and SVIs

A switch virtual interface (SVI) represents the Layer 3 interface of a VLAN. Layer 3 devices communicate with a private VLAN only through the primary VLAN and not through secondary VLANs. Configure Layer 3 VLAN interfaces (SVIs) only for primary VLANs. You cannot configure Layer 3 VLAN interfaces for secondary VLANs. SVIs for secondary VLANs are inactive while the VLAN is configured as a secondary VLAN.

- If you try to configure a VLAN with an active SVI as a secondary VLAN, the configuration is not allowed until you disable the SVI.
- If you try to create an SVI on a VLAN that is configured as a secondary VLAN and the secondary VLAN is already mapped at Layer 3, the SVI is not created, and an error is returned. If the SVI is not mapped at Layer 3, the SVI is created, but it is automatically shut down.

When the primary VLAN is associated with and mapped to the secondary VLAN, any configuration on the primary VLAN is propagated to the secondary VLAN SVIs. For example, if you assign an IP subnet to the primary VLAN SVI, this subnet is the IP subnet address of the entire private VLAN.

Private VLAN with Dynamic MAC Address

The MAC addresses learnt in the secondary VLAN are replicated to the primary VLAN and not vice-versa. This saves the hardware l2 cam space. The primary VLAN is always used for forwarding lookups in both directions.

Dynamic MAC addresses learned in Primary VLAN of a private VLAN are then, if required, replicated in the secondary VLANs. For example, if a MAC-address is dynamically received on the secondary VLAN, it will be learnt as part of primary VLAN. In case of isolated VLANs, a blocked entry for the same mac will be added to secondary VLAN in the mac address table. So, MAC learnt on host ports in secondary domain are installed as blocked type entries. All mac entries are learnt on secondary VLANs, even if the traffic ingresses from primary VLAN.

However, if a MAC-address is dynamically learnt in the primary VLAN it will not get replicated in the associated secondary VLANS.

Private VLAN with Static MAC Address

Users are not required to replicate the Static MAC Address CLI for private VLAN hosts as compare to legacy model.

Example:

- In the legacy model, if the user configures a static MAC address, they need to add the same static MAC address in the associated VLAN too. For example, if MAC address A is user configured on port 1/0/1

in VLAN 101, where VLAN 101 is a secondary VLAN and VLAN 100 is a primary VLAN, then the user has to configure

```
mac-address static A vlan 101 interface G1/1
mac-address static A vlan 100 interface G1/1
```

- In this device, the user does not need to replicate the mac address to the associated VLAN. For the above example, user has to configure only

```
mac-address static A vlan 101 interface G1/1
```

Private VLAN Interaction with VACL/QOS

Private VLANs are bidirectional in case of this device, as compared to "Unidirectional" in other platforms.

After layer-2 forward lookup, proper egress VLAN mapping happens and all the egress VLAN based feature processing happens in the egress VLAN context.

When a frame in Layer-2 is forwarded within a private VLAN, the VLAN map is applied at the ingress side and at the egress side. When a frame is routed from inside a private VLAN to an external port, the private-VLAN map is applied at the ingress side. Similarly, when the frame is routed from an external port to a Private VLAN, the private-VLAN is applied at the egress side. This is applicable to both bridged and routed traffic.

Bridging:

- For upstream traffic from secondary VLAN to primary VLAN, the MAP of the secondary VLAN is applied on the ingress side and the MAP of the primary VLAN is applied on the egress side.
- For downstream traffic from primary VLAN to secondary VLAN, the MAP of the primary VLAN is applied in the ingress direction and the MAP of the secondary VLAN is applied in the egress direction.

Routing

If we have two private VLAN domains - PV1 (sec1, prim1) and PV2 (sec2, prim2). For frames routed from PV1 to PV2:

- The MAP of sec1 and L3 ACL of prim1 is applied in the ingress port.
- The MAP of sec2 and L3 ACL of prim2 is applied in the egress port.

For packets going upstream or downstream from isolated host port to promiscuous port, the isolated VLAN's VACL is applied in the ingress direction and primary VLAN's VACL is applied in the egress direction. This allows user to configure different VACL for different secondary VLAN in a same primary VLAN domain.

Note 2-way community VLAN is now not required as the private VLANs on this device are always bi-directional.

Private-VLAN Configuration Guidelines

The following sections provide information about Private-VLAN configuration guidelines:

Default Private-VLAN Configurations

No private VLANs are configured.

Secondary and Primary VLAN Configuration

Follow these guidelines when configuring private VLANs:

- Private VLANs are supported in transparent mode for VTP 1, 2 and 3. If the device is running VTP version 1 or 2, you must set VTP to transparent mode. After you configure a private VLAN, you should not change the VTP mode to client or server. VTP version 3 supports private VLANs in all modes.
- With VTP version 1 or 2, after you have configured private VLANs, use the **copy running-config startup config** privileged EXEC command to save the VTP transparent mode configuration and private-VLAN configuration in the device startup configuration file. Otherwise, if the device resets, it defaults to VTP server mode, which does not support private VLANs. VTP version 3 does support private VLANs.
- VTP version 1 and 2 do not propagate private-VLAN configuration. You must configure private VLANs on each device where you want private-VLAN ports unless the devices are running VTP version 3, as VTP3 propagate private vlans.
- You cannot configure VLAN 1 or VLANs 1002 to 1005 as primary or secondary VLANs. Extended VLANs (VLAN IDs 1006 to 4094) can belong to private VLANs.
- A primary VLAN can have one isolated VLAN and multiple community VLANs associated with it. An isolated or community VLAN can have only one primary VLAN associated with it.
- Although a private VLAN contains more than one VLAN, only one Spanning Tree Protocol (STP) instance runs for the entire private VLAN. When a secondary VLAN is associated with the primary VLAN, the STP parameters of the primary VLAN are propagated to the secondary VLAN.
- When copying a PVLAN configuration from a tftp server and applying it on a running-config, the PVLAN association will not be formed. You will need to check and ensure that the primary VLAN is associated to all the secondary VLANs.

  You can also use **configure replace flash:config_file force** instead of **copy flash:config_file running-config**.
- You can enable DHCP snooping on private VLANs. When you enable DHCP snooping on the primary VLAN, it is propagated to the secondary VLANs. If you configure DHCP on a secondary VLAN, the configuration does not take effect if the primary VLAN is already configured.
- When you enable IP source guard on private-VLAN ports, you must enable DHCP snooping on the primary VLAN.
- We recommend that you prune the private VLANs from the trunks on devices that carry no traffic in the private VLANs.
- You can apply different quality of service (QoS) configurations to primary, isolated, and community VLANs.
- Note the following considerations for sticky ARP:
  - Sticky ARP entries are those learned on SVIs and Layer 3 interfaces. These entries do not age out.
  - The **ip sticky-arp** global configuration command is supported only on SVIs belonging to private VLANs.
  - The **ip sticky-arp** interface configuration command is only supported on:
    - Layer 3 interfaces
    - SVIs belonging to normal VLANs

- SVIs belonging to private VLANs

For more information about using the **ip sticky-arp** *global* configuration and the **ip sticky-arp interface** configuration commands, see the command reference for this release.

- You can configure VLAN maps on primary and secondary VLANs. However, we recommend that you configure the same VLAN maps on private-VLAN primary and secondary VLANs.
- PVLANs are bidirectional. They can be applied at both the ingress and egress sides.

  When a frame inLayer-2 is forwarded within a private VLAN, the VLAN map is applied at the ingress side and at the egress side. When a frame is routed from inside a private VLAN to an external port, the private-VLAN map is applied at the ingress side. Similarly, when the frame is routed from an external port to a Private VLAN, the private-VLAN is applied at the egress side.

  Bridging

  - For upstream traffic from secondary VLAN to primary VLAN, the MAP of the secondary VLAN is applied on the ingress side and the MAP of the primary VLAN is applied on the egress side.
  - For downstream traffic from primary VLAN to secondary VLAN, the MAP of the primary VLAN is applied in the ingress direction and the MAP of the secondary VLAN is applied in the egress direction.

  Routing

  If we have two private VLAN domains - PV1 (sec1, prim1) and PV2 (sec2, prim2). For frames routed from PV1 to PV2:

  - The MAP of sec1 and L3 ACL of prim1 is applied in the ingress port .
  - The MAP of sec1 and L3 ACL of prim2 is applied in the egress port.
  - For packets going upstream or downstream from isolated host port to promiscuous port, the isolated VLAN's VACL is applied in the ingress direction and primary VLAN'S VACL is applied in the egress direction. This allows user to configure different VACL for different secondary VLAN in a same primary VLAN domain.

  To filter out specific IP traffic for a private VLAN, you should apply the VLAN map to both the primary and secondary VLANs.
- You can apply router ACLs only on the primary-VLAN SVIs. The ACL is applied to both primary and secondary VLAN Layer 3 traffic.
- Although private VLANs provide host isolation at Layer 2, hosts can communicate with each other at Layer 3.
- Private VLANs support these Switched Port Analyzer (SPAN) features:
  - You can configure a private-VLAN port as a SPAN source port.
  - You can use VLAN-based SPAN (VSPAN) on primary, isolated, and community VLANs or use SPAN on only one VLAN to separately monitor egress or ingress traffic.

Private VLAN Port Configuration

Follow these guidelines when configuring private VLAN ports:

- Use only the private VLAN configuration commands to assign ports to primary, isolated, or community VLANs. Layer 2 access ports assigned to the VLANs that you configure as primary, isolated, or community VLANs are inactive while the VLAN is part of the private VLAN configuration. Layer 2 trunk interfaces remain in the STP forwarding state.
- Do not configure ports that belong to a PAgP or LACP EtherChannel as private VLAN ports. While a port is part of the private VLAN configuration, any EtherChannel configuration for it is inactive.
- Enable Port Fast and BPDU guard on isolated and community host ports to prevent STP loops due to misconfigurations and to speed up STP convergence. When enabled, STP applies the BPDU guard feature to all Port Fast-configured Layer 2 LAN ports. Do not enable Port Fast and BPDU guard on promiscuous ports.
- If you delete a VLAN used in the private VLAN configuration, the private VLAN ports associated with the VLAN become inactive.
- Private VLAN ports can be on different network devices if the devices are trunk-connected and the primary and secondary VLANs have not been removed from the trunk.

How to Configure Private VLANs

The following sections provide information about configuring Private VLANs:

Configuring Private VLANs

To configure a private VLAN, perform these steps:

Note Private vlans are supported in transparent mode for VTP 1, 2 and 3. Private VLANS are also supported on server mode with VTP 3.

Procedure

Step 1 Set VTP mode to **transparent**

Note
Note: For VTP3, you can set mode to either server or transparent mode.

Step 2 Create the primary and secondary VLANs and associate them.

See Configuring and Associating VLANs in a Private VLAN

Note
If the VLAN is not created already, the private-VLAN configuration process creates it.

Step 3 Configure interfaces to be isolated or community host ports, and assign VLAN membership to the host port.

See Configuring a Layer 2 Interface as a Private VLAN Host Port

Step 4 Configure interfaces as promiscuous ports, and map the promiscuous ports to the primary-secondary VLAN pair.

See Configuring a Layer 2 Interface as a Private VLAN Promiscuous Port

Step 5 If inter-VLAN routing will be used, configure the primary SVI, and map secondary VLANs to the primary.

See Mapping Secondary VLANs to a Primary VLAN Layer 3 VLAN Interface

Step 6 Verify private-VLAN configuration.

Configuring and Associating VLANs in a Private VLAN

The **private-vlan** commands do not take effect until you exit VLAN configuration mode.

To configure and associate VLANs in a Private VLAN, perform these steps:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vtp mode transparent** **Example:** `Device(config)# vtp mode transparent`	Sets VTP mode to transparent (disable VTP). **Note** For VTP3, you can set mode to either server or transparent mode
Step 4	**vlan** *vlan-id* **Example:** `Device(config)# vlan 20`	Enters VLAN configuration mode and designates or creates a VLAN that will be the primary VLAN. The VLAN ID range is 2 to 1001 and 1006 to 4094.
Step 5	**private-vlan primary** **Example:** `Device(config-vlan)# private-vlan`	Designates the VLAN as the primary VLAN.

	Command or Action	Purpose
	primary	
Step 6	**exit** **Example:** `Device(config-vlan)# exit`	Returns to global configuration mode.
Step 7	**vlan** *vlan-id* **Example:** `Device(config)# vlan 501`	(Optional) Enters VLAN configuration mode and designates or creates a VLAN that will be an isolated VLAN. The VLAN ID range is 2 to 1001 and 1006 to 4094.
Step 8	**private-vlan isolated** **Example:** `Device(config-vlan)# private-vlan isolated`	Designates the VLAN as an isolated VLAN.
Step 9	**exit** **Example:** `Device(config-vlan)# exit`	Returns to global configuration mode.
Step 10	**vlan** *vlan-id* **Example:** `Device(config)# vlan 502`	(Optional) Enters VLAN configuration mode and designates or creates a VLAN that will be a community VLAN. The VLAN ID range is 2 to 1001 and 1006 to 4094.
Step 11	**private-vlan community** **Example:** `Device(config-vlan)# private-vlan community`	Designates the VLAN as a community VLAN.
Step 12	**exit** **Example:** `Device(config-vlan)# exit`	Returns to global configuration mode.
Step 13	**vlan** *vlan-id* **Example:**	(Optional) Enters VLAN configuration mode and designates or creates a VLAN that will be

	Command or Action	Purpose
	`Device(config)# vlan 503`	a community VLAN. The VLAN ID range is 2 to 1001 and 1006 to 4094.
Step 14	**private-vlan community** **Example:** `Device(config-vlan)# private-vlan community`	Designates the VLAN as a community VLAN.
Step 15	**exit** **Example:** `Device(config-vlan)# exit`	Returns to global configuration mode.
Step 16	**vlan** *vlan-id* **Example:** `Device(config)# vlan 20`	Enters VLAN configuration mode for the primary VLAN designated in Step 4.
Step 17	**private-vlan association** [**add** | **remove**] *secondary_vlan_list* **Example:** `Device(config-vlan)# private-vlan association 501-503`	Associates the secondary VLANs with the primary VLAN. It can be a single private-VLAN ID or a hyphenated range of private-VLAN IDs. • The *secondary_vlan_list* parameter cannot contain spaces. It can contain multiple comma-separated items. Each item can be a single private-VLAN ID or a hyphenated range of private-VLAN IDs. • The *secondary_vlan_list* parameter can contain multiple community VLAN IDs but only one isolated VLAN ID. • Enter a *secondary_vlan_list*, or use the **add** keyword with a *secondary_vlan_list* to associate secondary VLANs with a primary VLAN. • Use the **remove** keyword with a *secondary_vlan_list* to clear the association between secondary VLANs and a primary VLAN. • The command does not take effect until you exit VLAN configuration mode.

	Command or Action	Purpose
Step 18	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 19	**show vlan private-vlan** [**type**] or **show interfaces status** **Example:** `Device# show vlan private-vlan`	Verifies the configuration.
Step 20	**copy running-config startup config** **Example:** `Device# copy running-config startup-config`	Saves your entries in the device startup configuration file.

Configuring a Layer 2 Interface as a Private VLAN Host Port

Follow these steps to configure a Layer 2 interface as a private-VLAN host port and to associate it with primary and secondary VLANs:

Note Isolated and community VLANs are both secondary VLANs.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:**	Enters interface configuration mode for the Layer 2 interface to be configured.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet1/2`	
Step 4	**switchport mode private-vlan host** **Example:** `Device(config-if)# switchport mode private-vlan host`	Configures the Layer 2 port as a private-VLAN host port.
Step 5	**switchport private-vlan host-association** *primary_vlan_id secondary_vlan_id* **Example:** `Device(config-if)# switchport private-vlan host-association 20 501`	Associates the Layer 2 port with a private VLAN. **Note** This is a required step to associate the PVLAN to a Layer 2 interface.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show interfaces** [*interface-id*] **switchport** **Example:** `Device# show interfaces gigabitethernet1/2 switchport`	Verifies the configuration.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Layer 2 Interface as a Private VLAN Promiscuous Port

Follow these steps to configure a Layer 2 interface as a private VLAN promiscuous port and map it to primary and secondary VLANs:

Note Isolated and community VLANs are both secondary VLANs.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/2`	Enters interface configuration mode for the Layer 2 interface to be configured.
Step 4	**switchport mode private-vlan promiscuous** **Example:** `Device(config-if)# switchport mode private-vlan promiscuous`	Configures the Layer 2 port as a private VLAN promiscuous port.
Step 5	**switchport private-vlan mapping** *primary_vlan_id* {**add** | **remove**} *secondary_vlan_list* **Example:** `Device(config-if)# switchport private-vlan mapping 20 add 501-503`	Maps the private VLAN promiscuous port to a primary VLAN and to selected secondary VLANs. • The *secondary_vlan_list* parameter cannot contain spaces. It can contain multiple comma-separated items. Each item can be a single private VLAN ID or a hyphenated range of private VLAN IDs. • Enter a *secondary_vlan_list*, or use the **add** keyword with a *secondary_vlan_list* to map the secondary VLANs to the private VLAN promiscuous port. • Use the **remove** keyword with a *secondary_vlan_list* to clear the mapping between secondary VLANs and the private VLAN promiscuous port.

	Command or Action	Purpose
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show interfaces** [*interface-id*] **switchport** **Example:** `Device# show interfaces gigabitethernet1/2 switchport`	Verifies the configuration.
Step 8	**copy running-config startup config** **Example:** `Device# copy running-config startup-config`	Saves your entries in the device startup configuration file.

Mapping Secondary VLANs to a Primary VLAN Layer 3 VLAN Interface

If the private VLAN will be used for inter-VLAN routing, you configure an SVI for the primary VLAN and map secondary VLANs to the SVI.

Note Isolated and community VLANs are both secondary VLANs.

Follow these steps to map secondary VLANs to the SVI of a primary VLAN to allow Layer 3 switching of private VLAN traffic:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface vlan** *primary_vlan_id* **Example:** `Device(config)# interface vlan 20`	Enters interface configuration mode for the primary VLAN, and configures the VLAN as an SVI. The VLAN ID range is 2 to 1001 and 1006 to 4094.
Step 4	**private-vlan mapping** [**add** | **remove**] *secondary_vlan_list* **Example:** `Device(config-if)# private-vlan mapping 501-503`	Maps the secondary VLANs to the Layer 3 VLAN interface of a primary VLAN to allow Layer 3 switching of private VLAN ingress traffic. **Note** The **private-vlan mapping** interface configuration command only affects private VLAN traffic that is Layer 3 switched. • The *secondary_vlan_list* parameter cannot contain spaces. It can contain multiple comma-separated items. Each item can be a single private-VLAN ID or a hyphenated range of private-VLAN IDs. • Enter a *secondary_vlan_list*, or use the **add** keyword with a *secondary_vlan_list* to map the secondary VLANs to a primary VLAN. • Use the **remove** keyword with a *secondary_vlan_list* to clear the mapping between secondary VLANs and a primary VLAN.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show interfaces private-vlan mapping** **Example:** `Device# show interfaces private-vlan mapping`	Verifies the configuration.
Step 7	**copy running-config startup config** **Example:** `Device# copy running-config startup-config`	Saves your entries in the device startup configuration file.

Monitoring Private VLANs

The following table displays the commands used to monitor private VLANs.

Table 36: Private VLAN Monitoring Commands

Command	Purpose
show interfaces status	Displays the status of interfaces, including the VLANs to which they belongs.
show vlan private-vlan [**type**]	
show interface switchport	Displays private VLAN configuration on interfaces.
show interface private-vlan mapping	Displays information about the private VLAN mapping for VLAN SVIs.

Configuration Examples for Private VLANs

This following sections provide configuration examples for Private VLANSs:

Example: Configuring and Associating VLANs in a Private VLAN

This example shows how to configure VLAN 20 as a primary VLAN, VLAN 501 as an isolated VLAN, and VLANs 502 and 503 as community VLANs, to associate them in a private VLAN, and to verify the configuration:

```
Device# configure terminal
Device(config)# vlan 20
Device(config-vlan)# private-vlan primary
Device(config-vlan)# exit
Device(config)# vlan 501
Device(config-vlan)# private-vlan isolated
Device(config-vlan)# exit
Device(config)# vlan 502
Device(config-vlan)# private-vlan community
Device(config-vlan)# exit
Device(config)# vlan 503
Device(config-vlan)# private-vlan community
Device(config-vlan)# exit
Device(config)# vlan 20
Device(config-vlan)# private-vlan association 501-503
Device(config-vlan)# end
Device# show vlan private-vlan
Primary   Secondary      Type
--------- -------------- -----------------
20        501            isolated
20        502            community
20        503            community
```

Example: Configuring an Interface as a Host Port

This example shows how to configure an interface as a private VLAN host port, associate it with a private VLAN pair, and verify the configuration:

```
Device# configure terminal
Device(config)# interface gigabitethernet1/1
Device(config-if)# switchport mode private-vlan host
Device(config-if)# switchport private-vlan host-association 20 501
Device(config-if)# end
Device# show interfaces gigabitethernet1/1 switchport
Name: Gi1/1
Switchport: Enabled
Administrative Mode: private-vlan host
Operational Mode: private-vlan host
Administrative Trunking Encapsulation: negotiate
Operational Trunking Encapsulation: native
Negotiation of Trunking: Off
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Voice VLAN: none
Administrative private-vlan host-association: 20 501
Administrative private-vlan mapping: none
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk private VLANs: none
Operational private-vlan:
20 501

<output truncated>
```

Example: Configuring an Interface as a Private VLAN Promiscuous Port

This example shows how to configure an interface as a private VLAN promiscuous port and map it to a private VLAN. The interface is a member of primary VLAN 20 and secondary VLANs 501 to 503 are mapped to it.

```
Device# configure terminal
Device(config)# interface gigabitethernet1/2
Device(config-if)# switchport mode private-vlan promiscuous
Device(config-if)# switchport private-vlan mapping 20 add 501-503
Device(config-if)# end
```

Use the **show vlan private-vlan** or the **show interface status** privileged EXEC command to display primary and secondary VLANs and private-VLAN ports on the device.

Example: Mapping Secondary VLANs to a Primary VLAN Interface

This example shows how to map the interfaces fo VLANs 501 and 502 to primary VLAN 10, which permits routing of secondary VLAN ingress traffic from private VLANs 501 and 502:

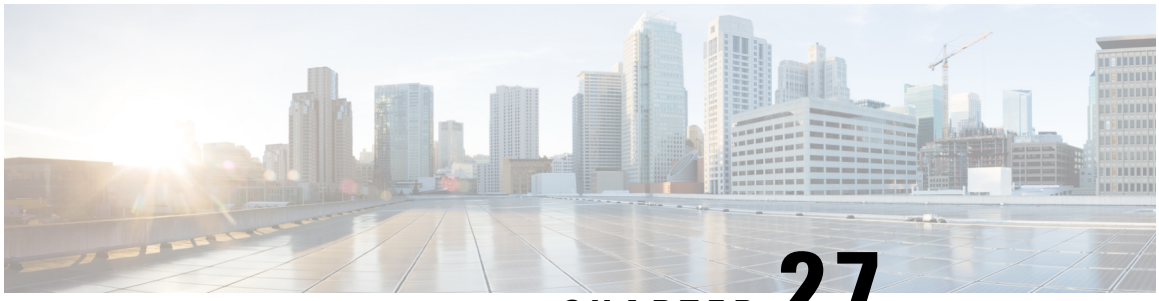
```
Device# configure terminal
Device(config)# interface vlan 20
```

```
Device(config-if)# private-vlan mapping 501-503
Device(config-if)# end
Device# show interfaces private-vlan mapping
Interface Secondary VLAN Type
--------- -------------- -----------------
vlan20    501            isolated
vlan20    502            community
vlan20    503            community
```

Example: Monitoring Private VLANs

This example shows output from the **show vlan private-vlan** command:

```
Device# show vlan private-vlan
Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------------
20      501       isolated          Gi1/1, Gi1/2
20      502       community         Gi1/2
20      503       community         Gi1/2
```



CHAPTER 27

Configuring Wired Dynamic PVLAN

The following sections provide information about configuring wired dynamic PVLAN:

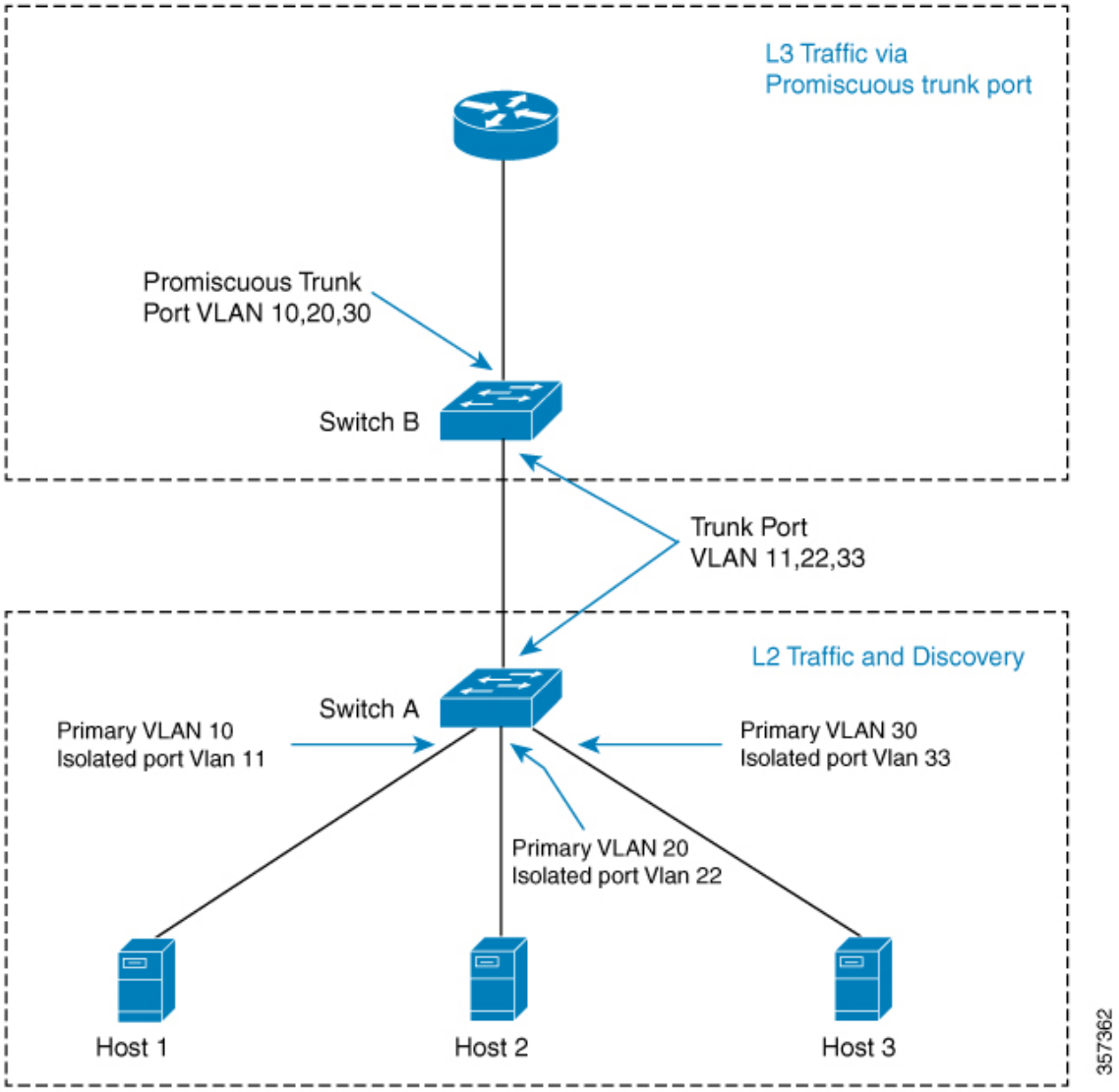
Restrictions for Wired Dynamic PVLAN

- High availability is not supported with Wired Dynamic PVLAN.
- Voice VLAN configuration cannot co-exist with this feature.
- Local Web Authentication (LWA) and Central Web Authentication (CWA) cannot be used with this feature.
- All wired clients using the dynamic PVLAN interface template will be programmed as data clients.
- Only interfaces with existing Access or PVLAN Host switchport mode support PVLAN template.
- Identity Based Networking Services 2.0 (IBNS 2.0) must be used for dynamic template support.

Information About Wired Dynamic PVLAN

Wired Dynamic PVLAN is a feature that uses a private VLAN with AAA authorization to isolate clients and provide Zero-Trust. It is a method to block peer to peer communications within a subnet/VLAN. Here, the client is assigned to a PVLAN which isolates a wired client connected on one port from all other ports on Layer 2 while the Layer 3 communication occurs via the promiscuous port. In this feature, a single wired data client is supported per port interface, to ensure point-to-point blocking.

Note Traffic from multiple clients on the same interface will not be blocked.



In this topology, the hosts are connected to Switch A and they can communicate only with the promiscuous trunk port on the switch. The PVLAN can be extended to span across multiple switches by adding intermediate switches. If there is a switch (Switch C) between Switch A and Switch B in the above topology, then layer 2 trunk ports need to be configured on the intermediate links. If case of a community VLAN, it allows packets to be seen on other hosts within the same community VLAN.

When a host is connected to a switch port with a cable, it is placed into an Isolated PVLAN where it cannot discover any other hosts. The host is then authenticated by the RADIUS server. Another scenario is when the port is placed in closed mode, and if the port is not authenticated, only Extensible Authentication Protocol over LAN (EAPoL) packets are allowed. Once the port is authenticated it is placed into an Isolated VLAN dynamically. As the host first authenticates with the RADIUS server, it sends the name of a dynamic interface template to be applied to the host's port. This interface template contains the configurations to enable the PVLAN Primary and Secondary VLANs on the port. With the template applied to the host, the switchport mode will be changed which will cause the port to flap from access mode to PVLAN mode.

Note The interface template with the same name as referred by AAA Authorization needs to be configured on the switch.

When the interface template is being applied, the port will physically go down for a time period set by the sticky timer and come up again. When the RADIUS server sends the interface template a second time, it is ignored as the conversion has been completed. The port is then assigned to a PVLAN which keeps it isolated. The host completes authorization and comes up to ready state.

Configure the keep time for which the interface template information is retained before it is removed from the port using the **access-session interface-template sticky timer** *time* command.

Configuring Wired Dynamic PVLAN

To configure Wired Dynamic PVLAN, perform these steps on the user device (Switch A in the above topology):

Before you begin

Ensure that the dot1x aaa is configured on the user device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan** *vlan-id* **Example:** `Device(config)# vlan 200`	(Optional) Enters VLAN configuration mode and designates or creates a VLAN that will be an isolated VLAN. The VLAN ID range is 2 to 1001 and 1006 to 4094.
Step 4	**private-vlan isolated** **Example:** `Device(config-vlan)# private-vlan isolated`	Designates the VLAN as an isolated VLAN.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-vlan)# exit`	Returns to global configuration mode.
Step 6	**vlan** *vlan-id* **Example:** `Device(config)# vlan 100`	Enters VLAN configuration mode and designates or creates a VLAN that will be the primary VLAN. The VLAN ID range is 2 to 1001 and 1006 to 4094.
Step 7	**private-vlan primary** **Example:** `Device(config-vlan)# private-vlan primary`	Designates the VLAN as the primary VLAN.
Step 8	**private-vlan association** [**add** | **remove**] *secondary_vlan_list* **Example:** `Device(config-vlan)# private-vlan association 200`	Associates the secondary VLANs with the primary VLAN. It can be a single private-VLAN ID or a hyphenated range of private-VLAN IDs. • The *secondary_vlan_list* parameter cannot contain spaces. It can contain multiple comma-separated items. Each item can be a single private-VLAN ID or a hyphenated range of private-VLAN IDs. • The *secondary_vlan_list* parameter can contain multiple community VLAN IDs but only one isolated VLAN ID. • Enter a *secondary_vlan_list*, or use the **add** keyword with a *secondary_vlan_list* to associate secondary VLANs with a primary VLAN. • Use the **remove** keyword with a *secondary_vlan_list* to clear the association between secondary VLANs and a primary VLAN. • The command does not take effect until you exit VLAN configuration mode.
Step 9	**exit** **Example:**	Returns to global configuration mode.

	Command or Action	Purpose
	`Device(config-vlan)# exit`	
Step 10	**template** *template-name* **Example:** `Device(config)# template PVLAN100_200_CFG`	Creates a user template and enters template configuration mode.
Step 11	**switchport mode private-vlan host** **Example:** `Device(config-template)# switchport mode private-vlan host`	Configures a Layer 2 port as a PVLAN host port on the template.
Step 12	**switchport private-vlan host-association** *primary_vlan_id secondary_vlan_id* **Example:** `Device(config-template)# switchport private-vlan host-association 100 200`	Configures the association of a Layer 2 port with a PVLAN on the template.
Step 13	**exit** **Example:** `Device(config-template)# exit`	Returns to global configuration mode.
Step 14	**access-session interface-template sticky timer** *time* **Example:** `Device(config)# access-session interface-template sticky timer 60`	Configures the keep time of the template globally. Once the last client leaves, the template will be removed from the port after the configured keep time. **Note** It is recommended that you set the sticky timer to 60 seconds.
Step 15	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet1/1`	Enters interface configuration mode and specifies the interface.
Step 16	**access-session interface-template sticky timer** *time*	Configures the keep time of the template on the interface. Once the last client leaves, the

	Command or Action	Purpose
	Example: `Device(config-if)# access-session interface-template sticky timer 60`	template will be removed from the port after the configured keep time. **Note** It is recommended that you set the sticky timer to 60 seconds.
Step 17	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

What to do next

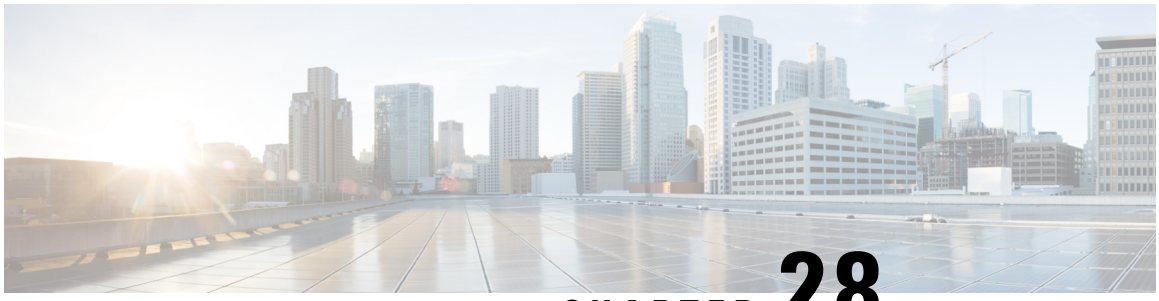
After the above steps, configure the Identity Services Engine (ISE) or any other RADIUS server to assign the template to the client's port interface after the client has been authenticated successfully.

If you are using the ISE, go to the **Policy** > **Policy Elements** > **Authorization** > **Authorization Profile** page. Check the **Interface Template** check box and enter the name of the template to be assigned to the client interface.

If you are using a different RADIUS server, the attribute **Cisco-AVpair="interface:template=name"** must be pushed to the switch after the initial client authentication has been completed.

PART III

Layer3 and Routing



CHAPTER 28

Configuring Bidirectional Forwarding Detection

Prerequisites for Bidirectional Forwarding Detection

- All participating switches must enable Cisco Express Forwarding and IP routing.
- Before BFD is deployed on a switch, it is necessary to configure one of the IP routing protocols that are supported by BFD. You should implement fast convergence for the routing protocol that you are using. See IP routing documentation for your version of Cisco IOS software for information on configuring fast convergence. See the "Restrictions for Bidirectional Forwarding Detection" section for more information on BFD routing protocol support in Cisco IOS software.

Restrictions for Bidirectional Forwarding Detection

- BFD works only for directly connected neighbors. BFD neighbors must be no more than one IP hop away. BFD does not support Multihop configurations.
- BFD support is not available for all platforms and interfaces. To confirm if a specific platform or interface has BFD support and to obtain the most accurate platform and hardware restrictions, see the Cisco IOS software release notes for your software version.
- The QoS policy for self-generated packets does not match BFD packets.
- The **class class-default** command matches BFD packets. So, you must make sure of the availability of appropriate bandwidth to prevent dropping of BFD packets due to oversubscription.
- BFD HA is not supported.
- When you use YANG operational models to delete individual BFD interval values, the whole BFD interval configuration gets deleted.

Information About Bidirectional Forwarding Detection

The following sections provide information about bidirectional forwarding detection.

BFD Operation

BFD provides a low-overhead, short-duration method of detecting failures in the forwarding path between two adjacent devices. These devices include the interfaces, data links, and forwarding planes.

BFD is a detection protocol that you enable at the interface and routing protocol levels. Cisco supports BFD asynchronous mode. BFD asychronous mode depends on the sending of BFD control packets between two systems to activate and maintain BFD neighbor sessions between devices. Therefore, in order to create a BFD session, you must configure BFD on both systems (or BFD peers). A BFD session is created once BFD is enabled on the interfaces and at the device level for the appropriate routing protocols. BFD timers are negotiated, and the BFD peers begin to send BFD control packets to each other at the negotiated interval.

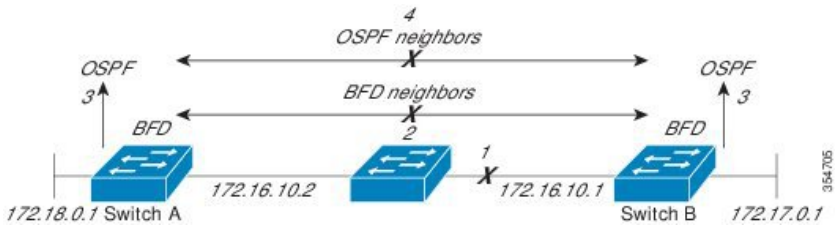
Neighbor Relationships

BFD provides fast BFD peer failure detection times independently. This is independent of all media types, encapsulations, topologies, and routing protocols such as BGP, EIGRP, IS-IS, and OSPF. BFD sends rapid failure detection notices to the routing protocols in the local device to initiate the routing table recalculation process. In this way, BFD contributes to greatly reduced overall network convergence time. The figure below shows a simple network with two devices running OSPF and BFD. When OSPF discovers a neighbor (1), it sends a request to the local BFD process. It initiates a BFD neighbor session with the OSPF neighbor device (2). The BFD neighbor session with the OSPF neighbor device is established (3).

Figure 36: BFD Process on a Network Configured with OSPF

The figure below shows what happens when a failure occurs in the network (1). The BFD neighbor session with the OSPF neighbor device is torn down (2). BFD notifies the local OSPF process that the BFD neighbor is no longer reachable (3). The local OSPF process tears down the OSPF neighbor relationship (4). If an alternative path is available, the devices immediately start converging on it.

Figure 37: BFD Process During a Network Failure



A routing protocol must register with BFD for every neighbor it acquires. Once a neighbor is registered, BFD initiates a session with the neighbor if a session does not already exist.

OSPF registers with BFD when:

- A neighbor finite state machine (FSM) transitions to full state.
- Both OSPF BFD and BFD are enabled.

On broadcast interfaces, OSPF establishes a BFD session only with the designated router (DR) and backup designated router (BDR). The session is not established between any two devices in a DROTHER state.

BFD Detection of Failures

Once a BFD session is established and timer negations are complete, BFD peers send BFD control packets. The packets act in the same manner as an IGP hello protocol to detect liveliness, except at a more accelerated rate. The following information should be noted:

- BFD is a forwarding path failure detection protocol. BFD detects a failure, but the routing protocol must act to bypass a failed peer.
- Cisco devices support BFD version 0. Devices use one BFD session for multiple client protocols in the implementation. For example, if a network is running OSPF and EIGRP across the same link to the same peer, only one BFD session is established. BFD shares session information with both routing protocols.

BFD Version Interoperability

All BFD sessions come up as Version 1 by default and are interoperable with Version 0. The system automatically performs BFD version detection, and BFD sessions between neighbors run in the highest common BFD version between neighbors. For example, if one BFD neighbor is running BFD Version 0 and the other BFD neighbor is running Version 1, the session runs BFD Version 0. The output from the **show bfd neighbors** [**details**] command verifies which BFD version a BFD neighbor is running.

See the "Example Configuring BFD in an EIGRP Network with Echo Mode Enabled by Default" for an example of BFD version detection.

BFD Session Limits

The maximum number of BFD sessions that can be created is 128.

BFD Support for Nonbroadcast Media Interfaces

The BFD feature is supported on routed, SVI, and L3 port channels. The **bfd interval** command must be configured on the interface to initiate BFD monitoring.

Benefits of Using BFD for Failure Detection

When you deploy any feature, it is important to consider all the alternatives and be aware of any trade-offs being made.

The closest alternative to BFD in conventional IS-IS, and OSPF deployments is the use of modified failure detection mechanisms for EIGRP, IS-IS, and OSPF routing protocols. If you use fast hellos for either IS-IS or OSPF, these Interior Gateway Protocol (IGP) protocols reduce their failure detection mechanisms to a minimum of one second.

There are several advantages to implementing BFD over reduced timer mechanisms for routing protocols:

- Although reducing the IS-IS, and OSPF timers can result in minimum detection timer of one to two seconds, BFD can provide failure detection in less than one second.
- Because BFD is not tied to any particular routing protocol, it can be used as a generic and consistent failure detection mechanism for IS-IS, and OSPF.
- Because some parts of BFD can be distributed to the data plane, it can be less CPU-intensive than the reduced IS-IS, and OSPF timers, which exist wholly at the control plane.

How to Configure Bidirectional Forwarding Detection

The following sections provide configurational information about bidirectional forwarding detection.

Configuring BFD Session Parameters on the Interface

To configure BFD on an interface, you must set the baseline BFD session parameters. Repeat the steps in this procedure for each interface over which you want to run BFD sessions to BFD neighbors.

The following procedure shows BFD configuration steps for a physical interface. Please use the corresponding BFD timer values for SVIs and ether-channels respectively.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	Perform one of the following steps: • **ip address** *ipv4-address mask* • **ipv6 address** *ipv6-address*/*mask* **Example:** Configuring an IPv4 address for the interface: `Device(config-if)#ip address 10.201.201.1 255.255.255.0` Configuring an IPv6 address for the interface:	Configures an IP address for the interface.

	Command or Action	Purpose
	`Device(config-if)#ipv6 address 2001:db8:1:1::1/32`	
Step 4	**bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier* **Example:** `Device(config-if)#bfd interval 100 min_rx 100 multiplier 3`	Enables BFD on the interface. The BFD interval configuration is removed when the subinterface on which it is configured is removed. The BFD interval configuration is not removed when: • An interface removes an IPv4 address. • An interface removes an IPv6 address is removed from an interface. • An interface disables IPv6. • An interface is shutdown • An interface globally or locally disables IPv4 CEF. • An interface globally or locally disables IPv6 CEF.
Step 5	**end** **Example:** `Device(config-if)#end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring BFD Support for Dynamic Routing Protocols

The following sections provide configurational information about BFD support for dynamic routing protocols.

Configuring BFD Support for IS-IS

This section describes the procedures for configuring BFD support for IS-IS so that IS-IS is a registered protocol with BFD and will receive forwarding path detection failure messages from BFD. There are two methods for enabling BFD support for IS-IS:

- You can enable BFD for all of the interfaces on which IS-IS is supporting IPv4 routing by using the **bfd all-interfaces** command in router configuration mode. You can then disable BFD for one or more of those interfaces using the **isis bfd disable** command in interface configuration mode.
- You can enable BFD for a subset of the interfaces for which IS-IS is routing by using the **isis bfd** command in interface configuration mode.

To configure BFD support for IS-IS, perform the steps in one of the following sections:

Prerequisites

- IS-IS must be running on all participating devices.
- The baseline parameters for BFD sessions on the interfaces that you want to run BFD sessions to BFD neighbors over must be configured. See the "Configuring BFD Session Parameters on the Interface" section for more information.

Note Output from the **show bfd neighbors details** command shows the configured intervals. The output does not show intervals that were changed because hardware-offloaded BFD sessions were configured with Tx and Rx intervals that are not multiples of 50 ms.

Configuring BFD Support for IS-IS for All Interfaces

To configure BFD on all IS-IS interfaces that support IPv4 routing, perform the steps in this section.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Device(config)#router isis tag1`	Specifies an IS-IS process and enters router configuration mode.
Step 4	**bfd all-interfaces** **Example:** `Device(config-router)#bfd all-interfaces`	Enables BFD globally on all interfaces that are associated with the IS-IS routing process.
Step 5	**exit** **Example:** `Device(config-router)#exit`	(Optional) Returns the device to global configuration mode.
Step 6	**interface** *type number* **Example:**	(Optional) Enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)#interface gigabitethernet 1/1`	
Step 7	**ip router isis** [*tag*] **Example:** `Device(config-if)#ip router isis tag1`	(Optional) Enables support for IPv4 routing on the interface.
Step 8	**isis bfd** [**disable**] **Example:** `Device(config-if)#isis bfd`	(Optional) Enables or disables BFD on a per-interface basis for one or more interfaces that are associated with the IS-IS routing process. **Note** You should use the **disable** keyword only if you had earlier enabled BFD on all the interfaces that IS-IS is associated with, using the **bfd all-interfaces** command in configuration mode.
Step 9	**end** **Example:** `Device(config-if)#end`	Exits interface configuration mode and returns the device to privileged EXEC mode.
Step 10	**show bfd neighbors** [**details**] **Example:** `Device#show bfd neighbors details`	(Optional) Displays information that can be used to verify if the BFD neighbor is active and displays the routing protocols that BFD has registered.
Step 11	**show clns interface** **Example:** `Device#show clns interface`	(Optional) Displays information that can be used to verify if BFD for IS-IS has been enabled for a specific IS-IS interface that is associated.

Configuring BFD Support for IS-IS for One or More Interfaces

To configure BFD for only one or more IS-IS interfaces, perform the steps in this section.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode.
Step 4	**ip router isis** [*tag*] **Example:** `Device(config-if)#ip router isis tag1`	Enables support for IPv4 routing on the interface.
Step 5	**isis bfd [disable]** **Example:** `Device(config-if)#isis bfd`	Enables or disables BFD on a per-interface basis for one or more interfaces that are associated with the IS-IS routing process. **Note** You should use the **disable** keyword only if you enabled BFD on all the interfaces that IS-IS is associated with using the **bfd all-interfaces** command in router configuration mode.
Step 6	**end** **Example:** `Device(config-if)#end`	Exits interface configuration mode and returns the device to privileged EXEC mode.
Step 7	**show bfd neighbors** [**details**] **Example:** `Device#show bfd neighbors details`	(Optional) Displays information that can help verify if the BFD neighbor is active and displays the routing protocols that BFD has registered.
Step 8	**show clns interface** **Example:** `Device#show clns interface`	(Optional) Displays information that can help verify if BFD for IS-IS has been enabled for a specific IS-IS interface that is associated.

Configuring BFD Support for OSPF

This section describes the procedures for configuring BFD support for OSPF so that OSPF is a registered protocol with BFD and will receive forwarding path detection failure messages from BFD. You can either configure BFD support for OSPF globally on all interfaces or configure it selectively on one or more interfaces.

There are two methods for enabling BFD support for OSPF:

- You can enable BFD for all the interfaces for which OSPF is routing by using the **bfd all-interfaces** command in router configuration mode. You can disable BFD support on individual interfaces using the **ip ospf bfd** [**disable**] command in interface configuration mode.
- You can enable BFD for a subset of the interfaces for which OSPF is routing by using the **ip ospf bfd** command in interface configuration mode.

See the following sections for tasks for configuring BFD support for OSPF:

Configuring BFD Support for OSPF for All Interfaces

To configure BFD for all OSPF interfaces, perform the steps in this section.

If you do not want to configure BFD on all OSPF interfaces and would rather configure BFD support specifically for one or more interfaces, see the "Configuring BFD Support for OSPF for One or More Interfaces" section.

Before you begin

- OSPF must be running on all participating devices.
- The baseline parameters for BFD sessions on the interfaces over which you want to run BFD sessions to BFD neighbors must be configured. See the "Configuring BFD Session Parameters on the Interface" section for more information.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**router ospf** *process-id* **Example:** `Device(config)#router ospf 4`	Specifies an OSPF process and enters router configuration mode.
Step 4	**bfd all-interfaces** **Example:** `Device(config-router)#bfd all-interfaces`	Enables BFD globally on all interfaces that are associated with the OSPF routing process.
Step 5	**exit** **Example:**	(Optional) Returns the device to global configuration mode. Enter this command only

	Command or Action	Purpose
	`Device(config-router)#exit`	if you want to perform Step 7 to disable BFD for one or more interfaces.
Step 6	**interface** *type number* **Example:** `Device(config)#interface gigabitethernet 1/1`	(Optional) Enters interface configuration mode. Enter this command only if you want to perform Step 7 to disable BFD for one or more interfaces.
Step 7	**ip ospf bfd** [**disable**] **Example:** `Device(config-if)#ip ospf bfd disable`	(Optional) Disables BFD on a per-interface basis for one or more interfaces that are associated with the OSPF routing process. **Note** You should use the **disable** keyword only if you enabled BFD on all the interfaces that OSPF is associated with using the **bfd all-interfaces** command in router configuration mode.
Step 8	**end** **Example:** `Device(config-if)#end`	Exits interface configuration mode and returns the router to privileged EXEC mode.
Step 9	**show bfd neighbors** [**details**] **Example:** `Device#show bfd neighbors detail`	(Optional) Displays information that can help verify if the BFD neighbor is active and displays the routing protocols that BFD has registered.
Step 10	**show ip ospf** **Example:** `Device#show ip ospf`	(Optional) Displays information that can help verify if BFD for OSPF has been enabled.

Configuring OSPF Support for BFD over IPv4 for One or More Interfaces

To configure BFD on one or more OSPF interfaces, perform the steps in this section.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode.
Step 4	**ip ospf bfd** [**disable**] **Example:** `Device(config-if)#ip ospf bfd`	Enables or disables BFD on a per-interface basis for one or more interfaces that are associated with the OSPF routing process. **Note** Use the **disable** keyword only if you enable BFD on all the interfaces that OSPF is associated with using the **bfd all-interfaces** command in router configuration mode.
Step 5	**end** **Example:** `Device(config-if)#end`	Exits interface configuration mode and returns the device to privileged EXEC mode.
Step 6	**show bfd neighbors** [**details**] **Example:** `Device#show bfd neighbors details`	(Optional) Displays information that can help verify if the BFD neighbor is active and displays the routing protocols that BFD has registered. **Note** If hardware-offloaded BFD sessions are configured with Tx and Rx intervals that are not multiples of 50 ms, the hardware intervals are changed. However, output from the **show bfd neighbors details** command displays only the configured intervals, not the interval values that change.
Step 7	**show ip ospf** **Example:** `Device#show ip ospf`	(Optional) Displays information that can help verify if BFD support for OSPF has been enabled.

Configuring BFD Support for HSRP

Perform this task to enable BFD support for Hot Standby Router Protocol (HSRP.) Repeat the steps in this procedure for each interface over which you want to run BFD sessions to HSRP peers.

HSRP supports BFD by default. If HSRP support for BFD has been manually disabled, you can reenable it at the device level to enable BFD support globally for all interfaces or on a per-interface basis at the interface level.

Before you begin

- HSRP must be running on all participating devices.
- Cisco Express Forwarding must be enabled.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**ip cef** [**distributed**] **Example:** `Device(config)#ip cef`	Enables Cisco Express Forwarding or distributed Cisco Express Forwarding.
Step 4	**interface** *type number* **Example:** `Device(config)#interface giabitethernet 1/1`	Enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Device(config-if)#ip address 10.1.0.22 255.255.0.0`	Configures an IP address for the interface.
Step 6	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)#standby 1 ip 10.0.0.11`	Activates HSRP.
Step 7	**standby bfd** **Example:**	(Optional) Enables HSRP support for BFD on the interface.

	Command or Action	Purpose
	`Device(config-if)#standby bfd`	
Step 8	**exit** **Example:** `Device(config-if)#exit`	Exits interface configuration mode.
Step 9	**standby bfd all-interfaces** **Example:** `Device(config)#standby bfd all-interfaces`	(Optional) Enables HSRP support for BFD on all interfaces.
Step 10	**exit** **Example:** `Device(config)#exit`	Exits global configuration mode.
Step 11	**show standby neighbors** **Example:** `Device#show standby neighbors`	(Optional) Displays information about HSRP support for BFD.

Configuring BFD Support for Static Routing

Perform this task to configure BFD support for static routing. Repeat the steps in this procedure on each BFD neighbor. For more information, see the "Example: Configuring BFD Support for Static Routing" section.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	Configures an interface and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)#interface GigabitEthernet1/1`	
Step 4	Perform one of the following steps: • **ip address** *ipv4-address mask* • **ipv6 address** *ipv6-address/mask* **Example:** Configuring an IPv4 address for the interface: `Device(config-if)#ip address 10.10.10.1 255.255.255.0` Configuring an IPv6 address for the interface: `Device(config-if)#ipv6 address 2001:db8:1:1::1/32`	Configures an IP address for the interface.
Step 5	**bfd interval** *milliseconds* **mix_rx** *milliseconds* **multiplier** *interval-multiplier* **Example:** `Device(config-if)#bfd interval 500 min_rx 500 multiplier 5`	Enables BFD on the interface. The bfd interval configuration is removed when the subinterface on which it is configured is removed. The bfd interval configuration is not removed when: • an IPv4 address is removed from an interface • an IPv6 address is removed from an interface • IPv6 is disabled from an interface. • an interface is shutdown • IPv4 CEF is disabled globally or locally on an interface. • IPv6 CEF is disabled globally or locally on an interface.
Step 6	**exit** **Example:** `Device(config-if)#exit`	Exits interface configuration mode and returns to global configuration mode.
Step 7	**ip route static bfd** *interface-type interface-number ip-address* [**group** *group-name* [**passive**]] **Example:**	Specifies a static route BFD neighbor. • The *interface-type*, *interface-number*, and *ip-address* arguments are required

	Command or Action	Purpose
	`Device(config)#ip route static bfd GigabitEthernet1/1 10.10.10.2`	because BFD support exists only for directly connected neighbors.
Step 8	**ip route** [**vrf** *vrf-name*] *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [**dhcp**] [*distance*] [**name** *next-hop-name*] [**permanent** | **track** *number*] [**tag** *tag*] **Example:** `Device(config)#ip route 10.0.0.0 255.0.0.0 GigabitEthernet1/1 10.10.10.2`	Specifies a static route BFD neighbor.
Step 9	**exit** **Example:** `Device(config)#exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 10	**show ip static route** **Example:** `Device#show ip static route`	(Optional) Displays static route database information.
Step 11	**show ip static route bfd** **Example:** `Device#show ip static route bfd`	(Optional) Displays information about the static BFD configuration from the configured BFD groups and nongroup entries.
Step 12	**exit** **Example:** `Device#exit`	Exits privileged EXEC mode and returns to user EXEC mode.

Configuring BFD Echo Mode

BFD echo mode is enabled by default, but you can disable it such that it can run independently in each direction.

BFD echo mode works with asynchronous BFD. Echo packets are sent by the forwarding engine and forwarded back along the same path in order to perform detection--the BFD session at the other end does not participate in the actual forwarding of the echo packets. The echo function and the forwarding engine are responsible for the detection process; therefore, the number of BFD control packets that are sent out between two BFD neighbors is reduced. In addition, because the forwarding engine is testing the forwarding path on the remote (neighbor) system without involving the remote system, there is an opportunity to improve the interpacket delay variance, thereby achieving quicker failure detection times than when using BFD Version 0 with BFD control packets for the BFD session.

Echo mode is described as without asymmetry when it is running on both sides (both BFD neighbors are running echo mode).

Prerequisites

- BFD must be running on all participating devices.
- Before using BFD echo mode, you must disable the sending of Internet Control Message Protocol (ICMP) redirect messages by entering the **no ip redirects** command, in order to avoid high CPU utilization.
- The baseline parameters for BFD sessions on the interfaces over which you want to run BFD sessions to BFD neighbors must be configured. See the Configuring BFD Session Parameters on the Interface section for more information.

Restrictions

BFD echo mode does not work with Unicast Reverse Path Forwarding (uRPF) configuration. If BFD echo mode and uRPF configurations are enabled, then the sessions will flap.

Disabling BFD Echo Mode Without Asymmetry

The steps in this procedure show how to disable BFD echo mode without asymmetry—no echo packets will be sent by the device, and the device will not forward BFD echo packets that are received from any neighbor devices.

Repeat the steps in this procedure for each BFD Device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**no bfd echo** **Example:** `Device(config)#no bfd echo`	Disables BFD echo mode. Use the **no** form to disable BFD echo mode.
Step 4	**end** **Example:** `Device(config)#end`	Exits global configuration mode and returns to privileged EXEC mode.

Creating and Configuring BFD Templates

You can configure a single-hop template to specify a set of BFD interval values. BFD interval values specified as part of the BFD template are not specific to a single interface.

Note Configuring BFD-template will disable echo mode.

Configuring a Single-Hop Template

Perform this task to create a BFD single-hop template and configure BFD interval timers.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**bfd-template single-hop** *template-name* **Example:** `Device(config)#bfd-template single-hop bfdtemplate1`	Creates a single-hop BFD template and enters BFD configuration mode.
Step 4	**interval min-tx** *milliseconds* **min-rx** *milliseconds* **multiplier** *multiplier-value* **Example:** `Device(bfd-config)#interval min-tx 120 min-rx 100 multiplier 3`	Configures the transmit and receive intervals between BFD packets, and specifies the number of consecutive BFD control packets that must be missed before BFD declares that a peer is unavailable.
Step 5	**end** **Example:** `Device(bfd-config)#end`	Exits BFD configuration mode and returns the device to privileged EXEC mode.

Monitoring and Troubleshooting BFD

This section describes how to retrieve BFD information for maintenance and troubleshooting. The commands in these tasks can be entered in any order as needed.

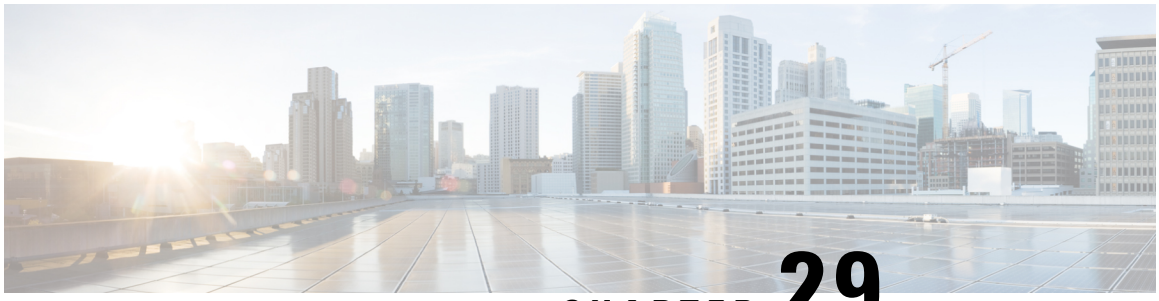
This section contains information for monitoring and troubleshooting BFD for the following Cisco platforms:

Monitoring and Troubleshooting BFD

To monitor or troubleshoot BFD, perform one or more of the steps in this section.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**show bfd neighbors** [**details**] **Example:** `Device#show bfd neighbors details`	(Optional) Displays the BFD adjacency database. The **details** keyword shows all BFD protocol parameters and timers per neighbor.
Step 3	**debug bfd** [**packet** | **event**] **Example:** `Device#debug bfd packet`	(Optional) Displays debugging information about BFD packets.



CHAPTER 29

Configuring BFD-EIGRP Support

Prerequisites for BFD-EIGRP Support

- Enhanced Interior Gateway Routing Protocol (EIGRP) must be running on all participating routers.
- The baseline parameters for Bidirectional Forwarding Detection (BFD) sessions on the interfaces over which you want to run BFD sessions to BFD neighbors must be configured using the **bfd** command.

Information About BFD-EIGRP Support

The BFD-EIGRP Support feature configures Bidirectional Forwarding Detection (BFD) feature for Enhanced Interior Gateway Routing Protocol (EIGRP) so that EIGRP registers with the BFD sessions on the routing interfaces, and receives forwarding path detection failure messages from BFD.

Use **bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier* command to enable BFD on any interface. Use the **bfd all-interfaces** command in router configuration mode to enable BFD for all of the interfaces where EIGRP routing is enabled. Use the **bfd interface** *type number* command in router configuration mode to enable BFD for a subset of the interfaces where EIGRP routing is enabled.

How to Configure BFD - EIGRP Support

To configure BFD-EIGRP support, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device>enable`	Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**router eigrp** *as-number* **Example:** `Device(config)#router eigrp 123`	Configures the EIGRP routing process and enters router configuration mode.
Step 4	Do one of the following: • **bfd all-interfaces** • **bfd interface** *type number* **Example:** `Device(config-router)#bfd all-interfaces` **Example:** `Device(config-router)#bfd interface gigabitethernet 1/1`	Enables BFD globally on all interfaces that are associated with the EIGRP routing process. or Enables BFD on a per-interface basis for one or more interfaces that are associated with the EIGRP routing process.
Step 5	**end** **Example:** `Device(config-router)#end`	Exits router configuration mode and returns the device to privileged EXEC mode.
Step 6	**show bfd neighbors** [**details**] **Example:** `Device#show bfd neighbors details`	(Optional) Verifies that the BFD neighbor is active and displays the routing protocols that BFD has registered.
Step 7	**show ip eigrp interfaces** [*type number*] [*as-number*] [**detail**] **Example:** `Device#show ip eigrp interfaces detail`	(Optional) Displays the interfaces for which BFD support for EIGRP has been enabled.

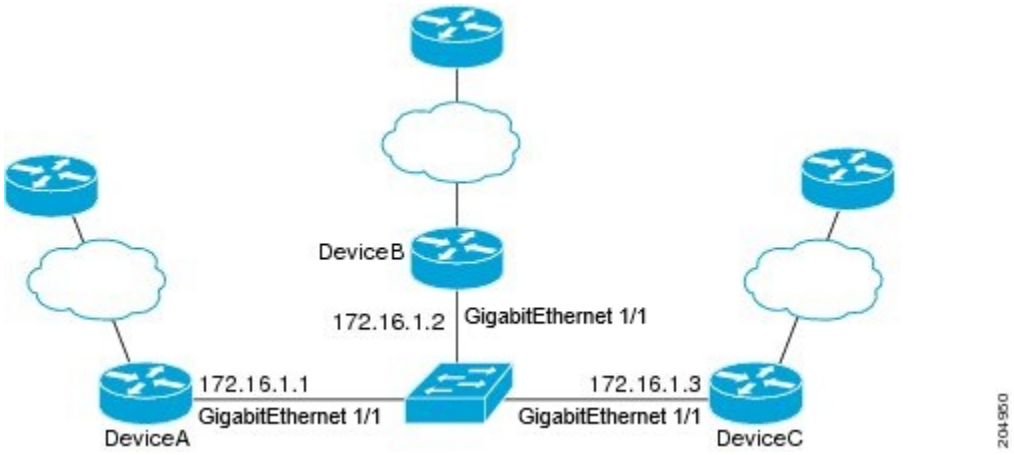
Configuration Example for BFD in an EIGRP Network with Echo Mode Enabled by Default

In the following example, the EIGRP network contains DeviceA, DeviceB, and DeviceC. Interface GigabitEthernet 1/1 on DeviceA is connected to the same network as Interface GigabitEthernet 1/2 on DeviceB. Interface GigabitEthernet 1/2 on DeviceB is connected to the same network as Interface GigabitEthernet 1/1 on DeviceC.

DeviceA and DeviceB are running BFD Version 1, which supports echo mode, and DeviceC is running BFD Version 0, which does not support echo mode. The BFD sessions between DeviceC and its BFD neighbors are said to be running echo mode with asymmetry because echo mode will run on the forwarding path for DeviceA and DeviceB, and their echo packets will return along the same path for BFD sessions and failure detections, while their BFD neighbor DeviceC runs BFD Version 0 and uses BFD controls packets for BFD sessions and failure detections.

The figure below shows a large EIGRP network with several devices, three of which are BFD neighbors that are running EIGRP as their routing protocol.

Figure 38: BFD Process on a Network Configured with EIGRP



The example, starting in global configuration mode, shows the configuration of BFD.

Configuration for DeviceA

```
interface gigabitethernet 1/1
ip address 172.16.1.1 255.255.255.0
bfd interval 50 min_rx 50 multiplier 3

duplex auto
speed auto
!
router eigrp 11
network 172.16.0.0
bfd all-interfaces
auto-summary
!
no ip http server
!
```

```
logging alarm informational
!
control-plane
!
line con 0
exec-timeout 30 0
stopbits 1
line aux 0
stopbits 1
line vty 0 4
login
!
!
end
```

Configuration for DeviceB

```
!
interface gigabitethernet 1/1

ip address 10.4.9.34 255.255.255.0
duplex auto
speed auto
!
iinterface gigabitethernet 1/2
ip address 172.16.1.2 255.255.255.0
bfd interval 50 min_rx 50 multiplier 3
no shtdown
duplex auto
speed auto
!
router eigrp 11
network 172.16.0.0
bfd all-interfaces
auto-summary
!
no ip http server
!
logging alarm informational
!
control-plane
!
line con 0
exec-timeout 30 0
stopbits 1
line aux 0
stopbits 1
line vty 0 4
login
!
!
end
```

Configuration for DeviceC

```
!
!
interface gigabitethernet 1/1

ip address 10.4.9.34 255.255.255.0
duplex auto
```

```
speed auto
!
interface gigabitethernet 1/2
ip address 172.16.1.2 255.255.255.0
bfd interval 50 min_rx 50 multiplier 3

duplex auto
speed auto
!
router eigrp 11
network 172.16.0.0
bfd all-interfaces
auto-summary
!

no ip http server
!
logging alarm informational
!
control-plane
!
line con 0
exec-timeout 30 0
stopbits 1
line aux 0
stopbits 1
line vty 0 4
login
!
!
end
```

The output from the **show bfd neighbors details** command from DeviceA verifies that BFD sessions are created among all three devices and that EIGRP is registered for BFD support. The first group of output shows that DeviceC with the IP address 172.16.1.3 runs BFD Version 0 and therefore does not use the echo mode. The second group of output shows that DeviceB with the IP address 172.16.1.2 runs BFD Version 1, and the 50 millisecond BFD interval parameter had been adopted. The relevant command output is shown in bold in the output.

```
DeviceA# show bfd neighbors details

OurAddr
       NeighAddr
      LD/RD  RH/RS    Holdown(mult)  State     Int
172.16.1.1    172.16.1.3
     5/3    1(RH)    150 (3 )       Up   Gi1/1
Session state is UP and not using echo function.
Local Diag: 0, Demand mode: 0, Poll bit: 0
MinTxInt: 50000, MinRxInt: 50000, Multiplier: 3
Received MinRxInt: 50000, Received Multiplier: 3
Holdown (hits): 150(0), Hello (hits): 50(1364284)
Rx Count: 1351813, Rx Interval (ms) min/max/avg: 28/64/49 last: 4 ms ago
Tx Count: 1364289, Tx Interval (ms) min/max/avg: 40/68/49 last: 32 ms ago
Registered protocols: EIGRP
Uptime: 18:42:45
Last packet: Version: 0
            - Diagnostic: 0
             I Hear You bit: 1     - Demand bit: 0
             Poll bit: 0           - Final bit: 0
             Multiplier: 3         - Length: 24
             My Discr.: 3          - Your Discr.: 5
```

```
            Min tx interval: 50000    - Min rx interval: 50000
            Min Echo interval: 0
OurAddr       NeighAddr
     LD/RD  RH/RS   Holdown(mult)  State     Int
172.16.1.1    172.16.1.2

    6/1    Up        0    (3 )   Up        Gi1/1
Session state is UP and using echo function with 50 ms interval.
Local Diag: 0, Demand mode: 0, Poll bit: 0
MinTxInt: 1000000, MinRxInt: 1000000, Multiplier: 3
Received MinRxInt: 1000000, Received Multiplier: 3
Holdown (hits): 3000(0), Hello (hits): 1000(317)
Rx Count: 305, Rx Interval (ms) min/max/avg: 1/1016/887 last: 448 ms ago
Tx Count: 319, Tx Interval (ms) min/max/avg: 1/1008/880 last: 532 ms ago
Registered protocols: EIGRP
Uptime: 00:04:30
Last packet: Version: 1

         - Diagnostic: 0
            State bit: Up         - Demand bit: 0
            Poll bit: 0           - Final bit: 0
            Multiplier: 3         - Length: 24
            My Discr.: 1          - Your Discr.: 6
            Min tx interval: 1000000    - Min rx interval: 1000000
            Min Echo interval: 50000
```

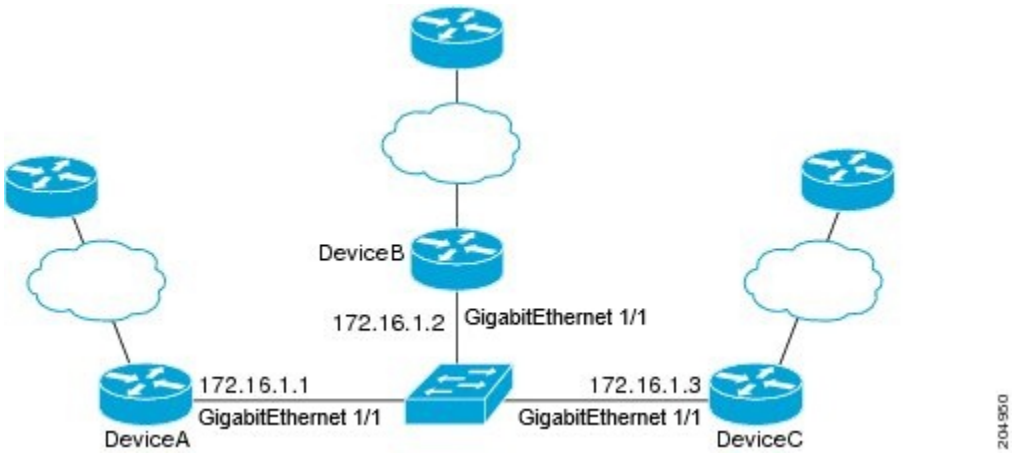
The output from the **show bfd neighbors details** command on DeviceB verifies that BFD sessions have been created and that EIGRP is registered for BFD support. As previously noted, DeviceA runs BFD Version 1, therefore echo mode is running, and DeviceC runs BFD Version 0, so echo mode does not run. The relevant command output is shown in bold in the output.

```
DeviceB# show bfd neighbors details

OurAddr       NeighAddr
     LD/RD  RH/RS   Holdown(mult)  State     Int
172.16.1.2    172.16.1.1
    1/6    Up        0    (3 )   Up        Gi1/1
Session state is UP and using echo function with 50 ms interval.
Local Diag: 0, Demand mode: 0, Poll bit: 0
MinTxInt: 1000000, MinRxInt: 1000000, Multiplier: 3
Received MinRxInt: 1000000, Received Multiplier: 3
Holdown (hits): 3000(0), Hello (hits): 1000(337)
Rx Count: 341, Rx Interval (ms) min/max/avg: 1/1008/882 last: 364 ms ago
Tx Count: 339, Tx Interval (ms) min/max/avg: 1/1016/886 last: 632 ms ago
Registered protocols: EIGRP
Uptime: 00:05:00
Last packet: Version: 1
           - Diagnostic: 0
             State bit: Up         - Demand bit: 0
             Poll bit: 0           - Final bit: 0
             Multiplier: 3         - Length: 24
             My Discr.: 6          - Your Discr.: 1
             Min tx interval: 1000000    - Min rx interval: 1000000
             Min Echo interval: 50000
OurAddr       NeighAddr

 LD/RD  RH/RS   Holdown(mult)  State     Int
172.16.1.2    172.16.1.3
     3/6    1(RH)     118  (3 )   Up        Gi1/1
Session state is UP and not using echo function.
Local Diag: 0, Demand mode: 0, Poll bit: 0
MinTxInt: 50000, MinRxInt: 50000, Multiplier: 3
```

```
Received MinRxInt: 50000, Received Multiplier: 3
Holdown (hits): 150(0), Hello (hits): 50(5735)
Rx Count: 5731, Rx Interval (ms) min/max/avg: 32/72/49 last: 32 ms ago
Tx Count: 5740, Tx Interval (ms) min/max/avg: 40/64/50 last: 44 ms ago
Registered protocols: EIGRP
Uptime: 00:04:45
Last packet: Version: 0
            - Diagnostic: 0
             I Hear You bit: 1     - Demand bit: 0
             Poll bit: 0           - Final bit: 0
             Multiplier: 3         - Length: 24
             My Discr.: 6          - Your Discr.: 3
             Min tx interval: 50000    - Min rx interval: 50000
             Min Echo interval: 0
```

The figure below shows that Interface 1/0 on DeviceB has failed. When Interface 1/0 on DeviceB is shut down, the BFD statistics of the corresponding BFD sessions on DeviceA and DeviceB are reduced.

Figure 39: BFD Process on Interfaces



When Interface GigabitEthernet 1/1 on DeviceB fails, BFD will no longer detect DeviceB as a BFD neighbor for DeviceA or for DeviceC. In this example, Interface GigabitEthernet 1/1 has been administratively shut down on DeviceB.

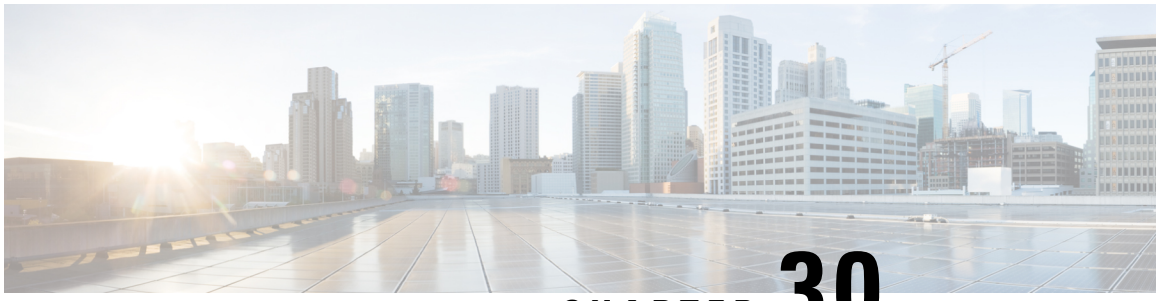
The following output from the **show bfd neighbors** command on DeviceA now shows only one BFD neighbor for DeviceA in the EIGRP network. The relevant command output is shown in bold in the output.

```
DeviceA# show bfd neighbors
OurAddr       NeighAddr

    LD/RD  RH/RS   Holdown(mult)  State     Int
172.16.1.1    172.16.1.3

    5/3    1(RH)     134   (3 )   Up        Gi1/1
```

The following output from the **show bfd neighbors** command on DeviceC also now shows only one BFD neighbor for DeviceC in the EIGRP network. The relevant command output is shown in bold in the output.

```
DeviceC# show bfd neighbors

OurAddr       NeighAddr

   LD/RD RH  Holdown(mult)  State     Int
```

```
172.16.1.3    172.16.1.1
    3/5  1   114  (3 )      Up        Gi1/1
```



CHAPTER 30

Configuring BFD Support for EIGRP IPv6

Prerequisites for BFD Support for EIGRP IPv6

EIGRP IPv6 sessions have a shutdown option in router, address family, and address-family interface configuration modes. To enable BFD support on EIGRP IPv6 sessions, the routing process should be in no shut mode in the above mentioned modes.

Restrictions for BFD Support for EIGRP IPv6

- The BFD Support for EIGRP IPv6 feature is supported only in EIGRP named mode.
- EIGRP supports only single-hop Bidirectional Forwarding Detection (BFD).
- The BFD Support for EIGRP IPv6 feature is not supported on passive interfaces.

Information About BFD Support for EIGRP IPv6

The BFD Support for EIGRP IPv6 feature provides Bidirectional Forwarding Detection (BFD) support for Enhanced Interior Gateway Routing Protocol (EIGRP) IPv6 sessions. It facilitates rapid fault detection and alternate-path selection in EIGRP IPv6 topologies. BFD is a detection protocol that provides a consistent failure-detection method for network administrators. Network administrators use BFD to detect forwarding path failures at a uniform rate and not at variable rates for 'Hello' mechanisms of different routing protocols. This failure-detection methodology ensures easy network profiling and planning and consistent and predictable reconvergence time. This document provides information about BFD support for EIGRP IPv6 networks and explains how to configure BFD support in EIGRP IPv6 networks.

How to Configure BFD Support for EIGRP IPv6

The following sections provide information on configuring BFD support for EIGRP IPv6 for an interface and all interfaces.

Configuring BFD Support on All Interfaces

The following steps show how to configure BFD support on all interfaces:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface Gigabitethernet1/1`	Specifies the interface type and number, and enters the interface configuration mode.
Step 5	**ipv6 address** *ipv6-address/prefix-length* **Example:** `Device(config-if)# ipv6 address 2001:DB8:A:B::1/64`	Configures an IPv6 address.
Step 6	**bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier* **Example:** `Device(config-if)# bfd interval 50 min_rx 50 multiplier 3`	Sets the baseline BFD session parameters on an interface.
Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 8	**router eigrp** *virtual-name* **Example:** `Device(config)# router eigrp name`	Specifies an EIGRP routing process and enters router configuration mode.
Step 9	**address-family ipv6 autonomous-system** *as-number* **Example:** `Device(config-router)# address-family ipv6 autonomous-system 3`	Enters address family configuration mode for IPv6 and configures an EIGRP routing instance.
Step 10	**eigrp router-id** *ip-address* **Example:** `Device(config-router-af)# eigrp router-id 172.16.1.3`	Sets the device ID used by EIGRP for this address family when EIGRP peers communicate with their neighbors.
Step 11	**af-interface default** **Example:** `Device(config-router-af)# af-interface default`	Configures interface-specific commands on all interfaces that belong to an address family in EIGRP named mode configurations. Enters address-family interface configuration mode.
Step 12	**bfd** **Example:** `Device(config-router-af-interface)# bfd`	Enables BFD on all interfaces.
Step 13	**End** **Example:** `Device(config-router-af-interface)# end`	Exits address-family interface configuration mode and returns to privileged EXEC mode.
Step 14	**show eigrp address-family ipv6 neighbors detail** **Example:** `Device# show eigrp address-family ipv6 neighbors detail`	(Optional) Displays detailed information about the neighbors that are discovered by EIGRP with BFD enabled on an interface.
Step 15	**show bfd neighbors** **Example:** `Device# show bfd neighbors`	(Optional) Displays BFD information to neighbors.

Configuring BFD Support on an Interface

The following steps show how to configure BFD support on an interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet1/1`	Specifies the interface type and number, and enters the interface configuration mode.
Step 5	**ipv6 address** *ipv6-address /prefix-length* **Example:** `Device(config-if)# ipv6 address 2001:DB8:A:B::1/64`	Configures an IPv6 address.
Step 6	**bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier* **Example:** `Device(config-if)# bfd interval 50 min_rx 50 multiplier 3`	Sets the baseline BFD session parameters on an interface.
Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	**router eigrp** *virtual-name* **Example:** `Device(config)# router eigrp name`	Specifies an EIGRP routing process and enters router configuration mode.
Step 9	**address-family ipv6 autonomous-system** *as-number* **Example:** `Device(config-router)# address-family ipv6 autonomous-system 3`	Enters address family configuration mode for IPv6 and configures an EIGRP routing instance.

	Command or Action	Purpose
Step 10	**eigrp router-id** *ip-address* **Example:** `Device(config-router-af)# eigrp router-id 172.16.1.3`	Sets the device ID used by EIGRP for this address family when EIGRP peers communicate with their neighbors.
Step 11	**af-interface** *interface-type interface-number* **Example:** `Device(config-router-af)# af-interface Gigabitethernet1/1`	Configures interface-specific commands on an interface that belongs to an address family in an EIGRP named mode configuration. Enters address-family interface configuration mode.
Step 12	**bfd** **Example:** `Device(config-router-af-interface)# bfd`	Enables BFD on the specified interface.
Step 13	**end** **Example:** `Device(config-router-af-interface)# end`	Exits address-family interface configuration mode and returns to privileged EXEC mode.
Step 14	**show eigrp address-family ipv6 neighbors** **Example:** `Device# show eigrp address-family ipv6 neighbors`	(Optional) Displays neighbors for which have BFD enabled.
Step 15	**show bfd neighbors** **Example:** `Device# show bfd neighbors`	(Optional) Displays BFD information to neighbors.

Configuration Examples for BFD Support for EIGRP IPv6

The following sections provide configuration examples for BFD support for EIGRP:

Example: Configuring BFD Support on All Interfaces

```
Device> enable
Device# configure terminal
Device(config)# ipv6 unicast-routing
Device(config)# interface GigabitEthernet1/1
Device(config-if)# ipv6 address 2001:0DB8:1::12/64
Device(config-if)# bfd interval 50 min_rx 50 multiplier 3
Device(config-if)# exit
Device(config)# router eigrp name
Device(config-router)# address-family ipv6 unicast autonomous-system 1
Device(config-router-af)# eigrp router-id 172.16.0.1
Device(config-router-af)# af-interface default
Device(config-router-af-interface)# bfd
Device(config-router-af-interface)# end
```

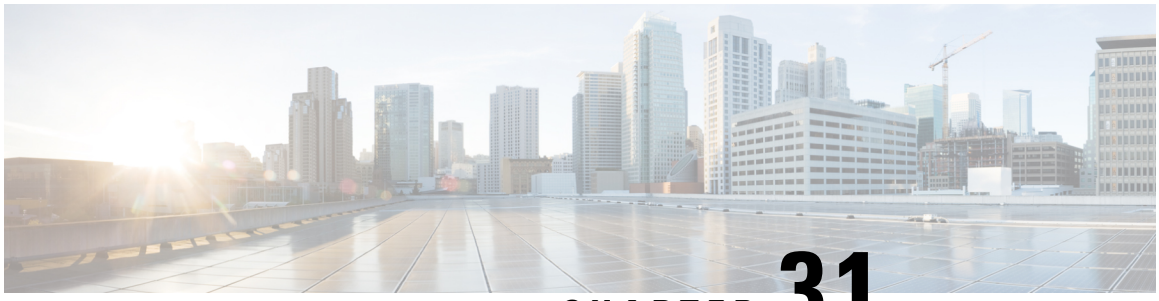
The following example displays the output for the **show eigrp address-family ipv6 neighbors detail** command.

```
Device# show eigrp address-family ipv6 neighbors detail
EIGRP-IPv6 VR(test) Address-Family Neighbors for AS(5)
H   Address                 Interface              Hold Uptime   SRTT   RTO  Q  Seq
                                                   (sec)         (ms)       Cnt Num
0   Link-local address:     Gi1/1                    14 00:02:04    1  4500  0  4
    GE80::10:2
   Version 23.0/2.0, Retrans: 2, Retries: 0, Prefixes: 1
   Topology-ids from peer - 0
   Topologies advertised to peer:   base

Max Nbrs: 0, Current Nbrs: 0

BFD sessions
NeighAddr         Interface
GE80::10:2            GigabitEthernet1/1
```

Example: Configuring BFD Support on an Interface

The following example shows how to configure BFD Support on an interface:

```
Device> enable
Device# configure terminal
Device(config)# ipv6 unicast-routing
Device(config)# interface GigabitEthernet1/1
Device(config-if)# ipv6 address 2001:DB8:A:B::1/64
Device(config-if)# bfd interval 50 min_rx 50 multiplier 3
Device(config-if)# exit
Device(config)# router eigrp name
Device(config-router)# address-family ipv6 autonomous-system 3
Device(config-router-af)# af-interface GigabitEthernet1/1
Device(config-router-af-interface)# bfd
Device(config-router-af-interface)# end
```



CHAPTER 31

IP Unicast Routing

This module describes how to configure IP Version 4 (IPv4) unicast routing on the switch.

Note In addition to IPv4 traffic, you can also enable IP Version 6 (IPv6) unicast routing and configure interfaces to forward IPv6 traffic .

Restrictions for IP Unicast Routing

- The switch does not support tunnel interfaces for unicast routed traffic.
- Subnetwork Access Protocol (SNAP) address resolution is not supported on this device.

IP Unicast Routing

This module describes how to configure IP Version 4 (IPv4) unicast routing on the switch.

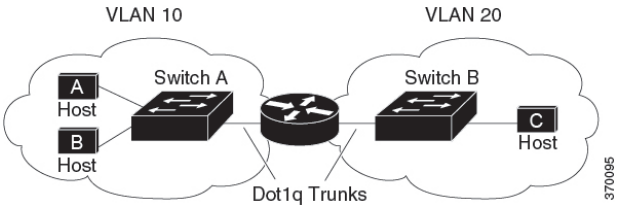
Note In addition to IPv4 traffic, you can also enable IP Version 6 (IPv6) unicast routing and configure interfaces to forward IPv6 traffic .

Information About IP Routing

In some network environments, VLANs are associated with individual networks or subnetworks. In an IP network, each subnetwork is mapped to an individual VLAN. Configuring VLANs helps control the size of the broadcast domain and keeps local traffic local. However, network devices in different VLANs cannot communicate with one another without a Layer 3 device (router) to route traffic between the VLAN, referred to as inter-VLAN routing. You configure one or more routers to route traffic to the appropriate destination VLAN.

This figure shows a basic routing topology. Switch A is in VLAN 10, and Switch B is in VLAN 20. The router has an interface in each VLAN.

Figure 40: Routing Topology Example



When Host A in VLAN 10 needs to communicate with Host B in VLAN 10, it sends a packet that is addressed to that host. Switch A forwards the packet directly to Host B, without sending it to the router.

When Host A sends a packet to Host C in VLAN 20, Switch A forwards the packet to the router, which receives the traffic on the VLAN 10 interface. The router checks the routing table, finds the correct outgoing interface, and forwards the packet on the VLAN 20 interface to Switch B. Switch B receives the packet and forwards it to Host C.

Types of Routing

Routers and Layer 3 switches can route packets in these ways:

- By using default routing
- By using preprogrammed static routes for the traffic

Default routing refers to sending traffic with a destination unknown to the router to a default outlet or destination.

Static unicast routing forwards packets from predetermined ports through a single path into and out of a network. Static routing is secure and uses little bandwidth, but does not automatically respond to changes in the network, such as link failures, and therefore, might result in unreachable destinations. As networks grow, static routing becomes a labor-intensive liability.

Dynamic routing protocols are used by routers to dynamically calculate the best route for forwarding traffic. There are two types of dynamic routing protocols:

- Routers using distance-vector protocols maintain routing tables with distance values of networked resources, and periodically pass these tables to their neighbors. Distance-vector protocols use one or a series of metrics for calculating the best routes. These protocols are easy to configure and use.
- Routers using link-state protocols maintain a complex database of network topology, based on the exchange of link-state advertisements (LSAs) between routers. LSAs are triggered by an event in the

network, which speeds up the convergence time or time that is required to respond to these changes. Link-state protocols respond quickly to topology changes, but require greater bandwidth and more resources than distance-vector protocols.

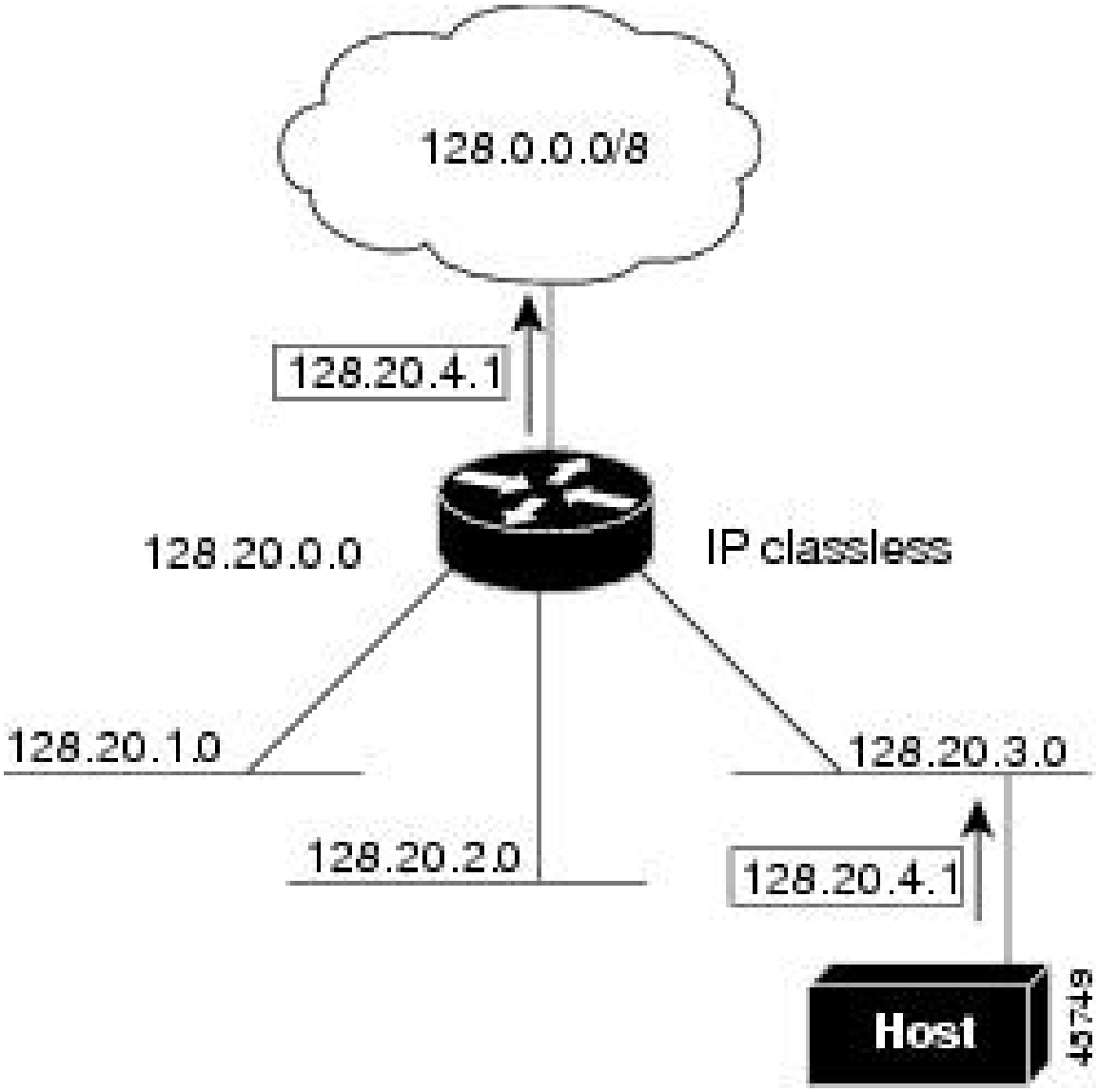
Distance-vector protocols that are supported by the switch are Routing Information Protocol (RIP), which uses a single distance metric (cost) to determine the best path. The switch also supports the Open Shortest Path First (OSPF) link-state protocol and Enhanced IGRP (EIGRP), which adds some link-state routing features to traditional Interior Gateway Routing Protocol (IGRP) to improve efficiency.

Classless Routing

By default, classless routing behavior is enabled on the device when it is configured to route. With classless routing, if a router receives packets for a subnet of a network with no default route, the router forwards the packet to the best supernet route. A supernet consists of contiguous blocks of Class C address spaces that are used to simulate a single, larger address space and is designed to relieve the pressure on the rapidly depleting Class B address space.

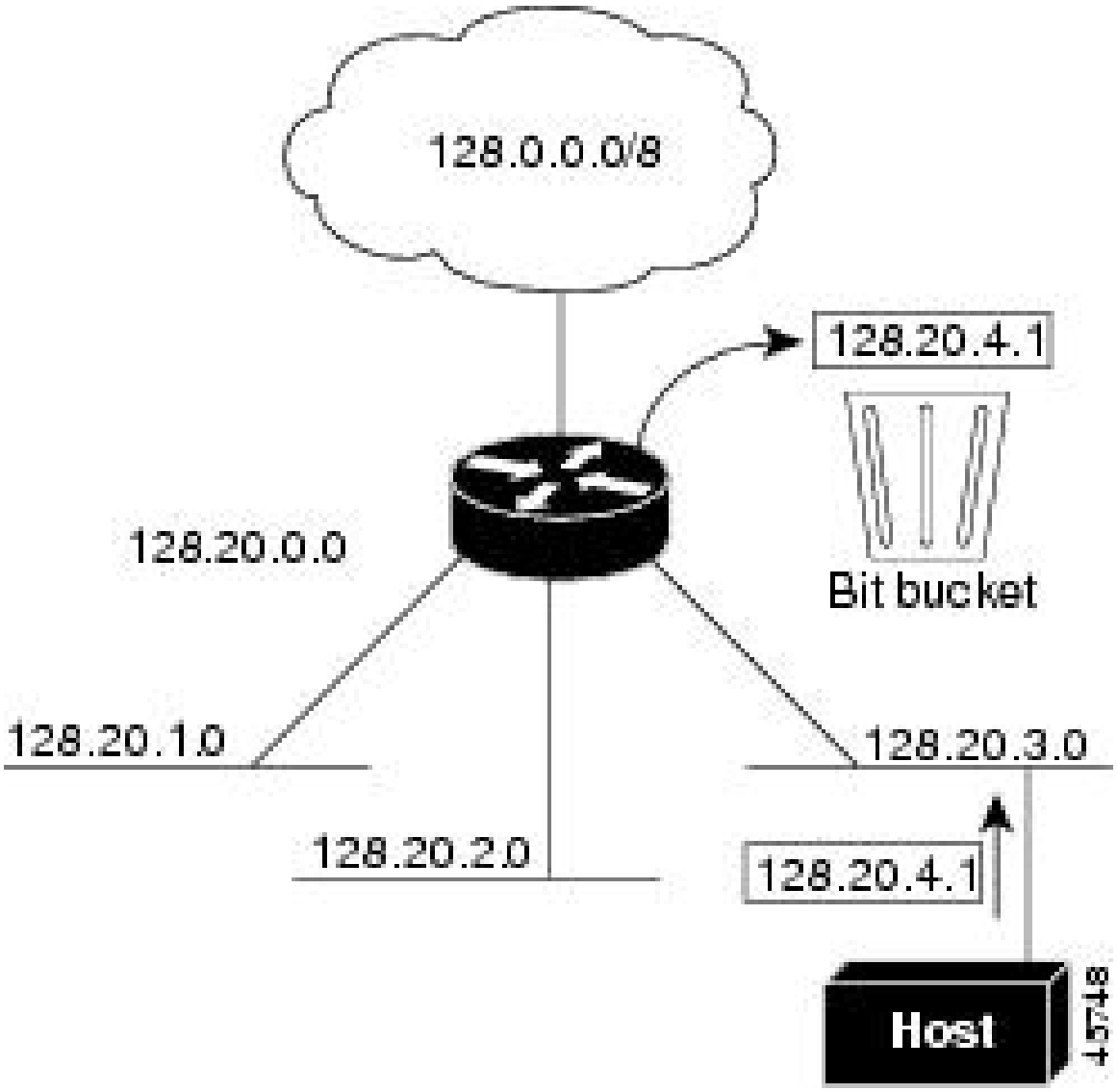
In the following figure, classless routing is enabled. When the host sends a packet to 120.20.4.1, instead of discarding the packet, the router forwards it to the best supernet route. If you disable classless routing and a router receives packets that are destined for a subnet of a network with no network default route, the router discards the packet.

Figure 41: IP Classless Routing



In the following figure, the router in network 128.20.0.0 is connected to subnets 128.20.1.0, 128.20.2.0, and 128.20.3.0. If the host sends a packet to 120.20.4.1, because there is no network default route, the router discards the packet.

Figure 42: No IP Classless Routing



To prevent the device from forwarding packets that are destined for unrecognized subnets to the best supernet route possible, you can disable classless routing behavior.

Address Resolution

You can control interface-specific handling of IP by using address resolution. A device using IP can have both a local address or MAC address, which uniquely defines the device on its local segment or LAN, and a network address, which identifies the network to which the device belongs.

The local address or MAC address is known as a data link address because it is contained in the data link layer (Layer 2) section of the packet header and is read by data link (Layer 2) devices. To communicate with a device on Ethernet, the software must learn the MAC address of the device. The process of learning the MAC address from an IP address is called *address resolution*. The process of learning the IP address from the MAC address is called *reverse address resolution*.

The device can use these forms of address resolution:

- Address Resolution Protocol (ARP) is used to associate IP address with MAC addresses. Taking an IP address as input, ARP learns the associated MAC address and then stores the IP address/MAC address association in an ARP cache for rapid retrieval. Then the IP datagram is encapsulated in a link-layer frame and sent over the network.
- Proxy ARP helps hosts with no routing tables learn the MAC addresses of hosts on other networks or subnets. If the device (router) receives an ARP request for a host that is not on the same interface as the ARP request sender, and if the router has all of its routes to the host through other interfaces, it generates a proxy ARP packet giving its own local data link address. The host that sent the ARP request then sends its packets to the router, which forwards them to the intended host.

The device also uses the Reverse Address Resolution Protocol (RARP), which functions the same as ARP does, except that the RARP packets request an IP address instead of a local MAC address. Using RARP requires a RARP server on the same network segment as the router interface. Use the **ip rarp-server** *address* interface configuration command to identify the server.

Proxy ARP

Proxy ARP, the most common method for learning about other routes, enables an Ethernet host with no routing information to communicate with hosts on other networks or subnets. The host assumes that all hosts are on the same local Ethernet and that they can use ARP to learn their MAC addresses. If a device receives an ARP request for a host that is not on the same network as the sender, the device evaluates whether it has the best route to that host. If it does, it sends an ARP reply packet with its own Ethernet MAC address, and the host that sent the request sends the packet to the device, which forwards it to the intended host. Proxy ARP treats all networks as if they are local, and performs ARP requests for every IP address.

ICMP Router Discovery Protocol

Router discovery allows the device to dynamically learn about routes to other networks using ICMP router discovery protocol (IRDP). IRDP allows hosts to locate routers. When operating as a client, the device generates router discovery packets. When operating as a host, the device receives router discovery packets. The device can also listen to Routing Information Protocol (RIP) routing updates and use this information to infer locations of routers. The device does not actually store the routing tables that are sent by routing devices; it merely keeps track of which systems are sending the data. The advantage of using IRDP is that it allows each router to specify both a priority and the time after which a device is assumed to be down if no further packets are received.

Each device that is discovered becomes a candidate for the default router, and a new highest-priority router is selected when a higher priority router is discovered, when the current default router is declared down, or when a TCP connection is about to time out because of excessive retransmissions.

IRDP packets are not sent while enabling or disabling IP routing. When interface is shutting down, the last IRDP message does not have a lifetime; it is 0 for all routers.

UDP Broadcast Packets and Protocols

User Datagram Protocol (UDP) is an IP host-to-host layer protocol, as is TCP. UDP provides a low-overhead, connectionless session between two end systems and does not provide for acknowledgment of received datagrams. Network hosts occasionally use UDP broadcasts to find address, configuration, and name information. If such a host is on a network segment that does not include a server, UDP broadcasts are normally

not forwarded. You can remedy this situation by configuring an interface on a router to forward certain classes of broadcasts to a helper address. You can use more than one helper address per interface.

You can specify a UDP destination port to control which UDP services are forwarded. You can specify multiple UDP protocols. You can also specify the Network Disk (ND) protocol, which is used by older diskless Sun workstations and the network security protocol SDNS.

By default, both UDP, and ND forwarding are enabled if a helper address has been defined for an interface.

Broadcast Packet Handling

After configuring an IP interface address, you can enable routing and configure one or more routing protocols, or you can configure the way that the device responds to network broadcasts. A broadcast is a data packet that is destined for all hosts on a physical network. The device supports two kinds of broadcasting:

- A directed broadcast packet is sent to a specific network or series of networks. A directed broadcast address includes the network or subnet fields.
- A flooded broadcast packet is sent to every network.

Note You can also limit broadcast, unicast, and multicast traffic on Layer 2 interfaces by using the **storm-control** interface configuration command to set traffic suppression levels.

Routers provide some protection from broadcast storms by limiting their extent to the local cable. Bridges (including intelligent bridges), because they are Layer 2 devices, forward broadcasts to all network segments, thus propagating broadcast storms. The best solution to the broadcast storm problem is to use a single broadcast address scheme on a network. In most modern IP implementations, you can set the address to be used as the broadcast address. Many implementations, including the one in the device, support several addressing schemes for forwarding broadcast messages.

IP Broadcast Flooding

You can allow IP broadcasts to be flooded throughout your internetwork in a controlled fashion by using the database created by the bridging STP. Using this feature also prevents loops. To support this capability, bridging must be configured on each interface that is to participate in the flooding. If bridging is not configured on an interface, it still can receive broadcasts. However, the interface never forwards broadcasts it receives, and the router never uses that interface to send broadcasts received on a different interface.

Packets that are forwarded to a single network address using the IP helper-address mechanism can be flooded. Only one copy of the packet is sent on each network segment.

To be considered for flooding, packets must meet these criteria. (Note that these are the same conditions used to consider packet forwarding using IP helper addresses.)

- The packet must be a MAC-level broadcast.
- The packet must be an IP-level broadcast.
- The packet must be a TFTP, DNS, Time, NetBIOS, ND, or BOOTP packet, or a UDP specified by the **ip forward-protocol udp** global configuration command.

- The time-to-live (TTL) value of the packet must be at least two.

A flooded UDP datagram is given the destination address specified with the **ip broadcast-address** interface configuration command on the output interface. The destination address can be set to any address. Thus, the destination address might change as the datagram propagates through the network. The source address is never changed. The TTL value is decremented.

When a flooded UDP datagram is sent out an interface (and the destination address possibly changed), the datagram is handed to the normal IP output routines and is, therefore, subject to access lists, if they are present on the output interface.

In the switch, the majority of packets are forwarded in hardware; most packets do not go through the switch CPU. For those packets that do go to the CPU, you can speed up spanning tree-based UDP flooding by a factor of about four to five times by using turbo-flooding. This feature is supported over Ethernet interfaces configured for ARP encapsulation.

Configuration Guidelines for IP Routing

By default, IP routing is disabled on the device, and you must enable it before routing can take place.

In the following procedures, the specified interface must be one of these Layer 3 interfaces:

- A routed port: a physical port configured as a Layer 3 port by using the **no switchport** interface configuration command.
- A switch virtual interface (SVI): a VLAN interface that is created by using the **interface vlan** *vlan_id* global configuration command and by default a Layer 3 interface.
- An EtherChannel port channel in Layer 3 mode: a port-channel logical interface that is created by using the **interface port-channel** *port-channel-number* global configuration command and binding the Ethernet interface into the channel group.

All Layer 3 interfaces on which routing will occur must have IP addresses assigned to them.

Note A Layer 3 switch can have an IP address that is assigned to each routed port and SVI.

Configuring routing consists of several main procedures:

- To support VLAN interfaces, create and configure VLANs on the switch and assign VLAN membership to Layer 2 interfaces. For more information, see the "Configuring VLANs" chapter.
- Configure Layer 3 interfaces.
- Enable IP routing on the switch.
- Assign IP addresses to the Layer 3 interfaces.
- Enable selected routing protocols on the switch.
- Configure routing protocol parameters (optional).

How to Configure IP Addressing

A required task for configuring IP routing is to assign IP addresses to Layer 3 network interfaces to enable the interfaces and allow communication with the hosts on those interfaces that use IP. The following sections describe how to configure various IP addressing features. Assigning IP addresses to the interface is required; the other procedures are optional.

Default IP Addressing Configuration

Table 37: Default Addressing Configuration

Feature	Default Setting
IP address	None defined.
ARP	No permanent entries in the Address Resolution Protocol (ARP) cache. Encapsulation: Standard Ethernet-style ARP. Timeout: 14400 seconds (4 hours).
IP broadcast address	255.255.255.255 (all ones).
IP classless routing	Enabled.
IP default gateway	Disabled.
IP directed broadcast	Disabled (all IP directed broadcasts are dropped).
IP domain	Domain list: No domain names defined. Domain lookup: Enabled. Domain name: Enabled.
IP forward-protocol	If a helper address is defined or User Datagram Protocol (UDP) flooding is configured, UDP forwarding is enabled on default ports. Any-local-broadcast: Disabled. Spanning Tree Protocol (STP): Disabled. Turbo-flood: Disabled.
IP helper address	Disabled.
IP host	Disabled.

Feature	Default Setting
IRDP	Disabled. Defaults when enabled: • Broadcast IRDP advertisements. • Maximum interval between advertisements: 600 seconds. • Minimum interval between advertisements: 0.75 times max interval • Preference: 0.
IP proxy ARP	Enabled.
IP routing	Disabled.
IP subnet-zero	Disabled.

Assigning IP Addresses to Network Interfaces

An IP address identifies a location to which IP packets can be sent. Some IP addresses are reserved for special uses and cannot be used for host, subnet, or network addresses. RFC 1166, “Internet Numbers,” contains the official description of IP addresses.

An interface can have one primary IP address. A mask identifies the bits that denote the network number in an IP address. When you use the mask to subnet a network, the mask is referred to as a subnet mask. To receive an assigned network number, contact your Internet service provider.

To assign IP addresses to network interfaces, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:**	Enters interface configuration mode, and specifies the Layer 3 interface to configure.

	Command or Action	Purpose
	`Device(config)#interface gigabitethernet 1/1`	
Step 4	**no switchport** **Example:** `Device(config-if)#no switchport`	Removes the interface from Layer 2 configuration mode (if it is a physical interface).
Step 5	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)#ip address 10.1.5.1 255.255.255.0`	Configures the IP address and IP subnet mask.
Step 6	**no shutdown** **Example:** `Device(config-if)#no shutdown`	Enables the physical interface.
Step 7	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 8	**show ip route** **Example:** `Device#show ip route`	Verifies your entries.
Step 9	**show ip interface** [*interface-id*] **Example:** `Device#show ip interface gigabitethernet 1/1`	Verifies your entries.
Step 10	**show running-config** **Example:** `Device#show running-config`	Verifies your entries.
Step 11	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Using Subnet Zero

Subnetting with a subnet address of zero is strongly discouraged because of the problems that can arise if a network and a subnet have the same addresses. For example, if network 131.108.0.0 is subnetted as 255.255.255.0, subnet zero would be written as 131.108.0.0, which is the same as the network address.

You can use the all ones subnet (131.108.255.0) and even though it is discouraged, you can enable the use of subnet zero if you need the entire subnet space for your IP address.

Use the **no ip subnet-zero** global configuration command to restore the default and disable the use of subnet zero.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**ip subnet-zero** **Example:** `Device(config)#ip subnet-zero`	Enables the use of subnet zero for interface addresses and routing updates.
Step 4	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device#show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Disabling Classless Routing

To prevent the device from forwarding packets that are destined for unrecognized subnets to the best supernet route possible, you can disable classless routing behavior.

To disable classless routing, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**no ip classless** **Example:** `Device(config)#no ip classless`	Disables classless routing behavior.
Step 4	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device#show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Address Resolution Methods

You can perform the following tasks to configure address resolution.

Defining a Static ARP Cache

ARP and other address resolution protocols provide dynamic mapping between IP addresses and MAC addresses. Because most hosts support dynamic address resolution, you usually do not need to specify static ARP cache entries. If you must define a static ARP cache entry, you can do so globally, which installs a permanent entry in the ARP cache that the device uses to translate IP addresses into MAC addresses. Optionally, you can also specify that the device responds to ARP requests as if it were the owner of the specified IP address. If you do not want the ARP entry to be permanent, you can specify a timeout period for the ARP entry.

To define a static arp cache, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**arp** *ip-address hardware-address type* **Example:** `Device(config)#ip 10.1.5.1 c2f3.220a.12f4 arpa`	Associates an IP address with a MAC (hardware) address in the ARP cache, and specifies encapsulation type as one of these: • **arpa**—ARP encapsulation for Ethernet interfaces • **sap**—HP's ARP type
Step 4	**arp** *ip-address hardware-address type* [**alias**] **Example:** `Device(config)#ip 10.1.5.3 d7f3.220d.12f5 arpa alias`	(Optional) Specifies that the switch responds to ARP requests as if it were the owner of the specified IP address.
Step 5	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the interface to configure.
Step 6	**arp timeout** *seconds* **Example:**	(Optional) Sets the length of time an ARP cache entry stays in the cache. The recommended value of ARP timeout is 4 hours

	Command or Action	Purpose
	`Device(config-if)#arp timeout 20000`	which is also the default setting. However, if your network experiences regular updates to ARP cache entries, consider changing the timeout. Note that decreasing the ARP timeout can result in increased network traffic. It is not recommended to set the ARP timeout to 60 seconds or less.
Step 7	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 8	**show interfaces** [*interface-id*] **Example:** `Device#show interfaces gigabitethernet 1/1`	Verifies the type of ARP and the timeout value that is used on all interfaces or a specific interface.
Step 9	**show arp** **Example:** `Device#show arp`	Views the contents of the ARP cache.
Step 10	**show ip arp** **Example:** `Device#show ip arp`	Views the contents of the ARP cache.
Step 11	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Setting ARP Encapsulation

By default, Ethernet ARP encapsulation (represented by the **arpa** keyword) is enabled on an IP interface.

To setting ARP encapsulation, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
	`Device>enable`	
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 4	**arp arpa** **Example:** `Device(config-if)#arp arpa`	Specifies the ARP encapsulation method. Use the **no arp arpa** command to disable ARP encapsulation method.
Step 5	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 6	**show interfaces** [*interface-id*] **Example:** `Device#show interfaces`	Verifies ARP encapsulation configuration on all interfaces or the specified interface.
Step 7	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Enabling Proxy ARP

By default, the device uses proxy ARP to help hosts learn MAC addresses of hosts on other networks or subnets.

To enable proxy ARP, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 4	**ip proxy-arp** **Example:** `Device(config-if)#ip proxy-arp`	Enables proxy ARP on the interface.
Step 5	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 6	**show ip interface** [*interface-id*] **Example:** `Device#show ip interface gigabitethernet 1/1`	Verifies the configuration on the interface or all interfaces.
Step 7	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Routing Assistance When IP Routing is Disabled

These mechanisms allow the device to learn about routes to other networks when it does not have IP routing that is enabled:

- Proxy ARP
- Default Gateway
- ICMP Router Discovery Protocol (IRDP)

Proxy ARP

Proxy ARP is enabled by default. To enable it after it has been disabled, see the "Enabling Proxy ARP" section. Proxy ARP works as long as other routers support it.

Configuring Default Gateway

Another method for locating routes is to define a default router or default gateway. All non-local packets are sent to this router, which either routes them appropriately or sends an IP Control Message Protocol (ICMP) redirect message back, defining which local router the host should use. The device caches the redirect messages and forwards each packet as efficiently as possible. A limitation of this method is that there is no means of detecting when the default router has gone down or is unavailable.

To configure default gateway, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**ip default-gateway** *ip-address* **Example:** `Device(config)#ip default gateway 10.1.5.1`	Sets up a default gateway (router).
Step 4	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)#end`	
Step 5	**show ip redirects** **Example:** `Device#show ip redirects`	Displays the address of the default gateway router to verify the setting.
Step 6	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring ICMP Router Discovery Protocol

The only required task for IRDP routing on an interface is to enable IRDP processing on that interface. When enabled, the default parameters apply.

You can optionally change any of these parameters. If you change the **maxadvertinterval** value, the **holdtime** and **minadvertinterval** values also change, so it is important to first change the **maxadvertinterval** value, before manually changing either the **holdtime** or **minadvertinterval** values.

To configure ICMP router discovery protocol, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.

	Command or Action	Purpose
Step 4	**ip irdp** **Example:** `Device(config-if)#ip irdp`	Enables IRDP processing on the interface.
Step 5	**ip irdp multicast** **Example:** `Device(config-if)#ip irdp multicast`	(Optional) Sends IRDP advertisements to the multicast address (224.0.0.1) instead of IP broadcasts. **Note** This command allows for compatibility with Sun Microsystems Solaris, which requires IRDP packets to be sent out as multicasts. Many implementations cannot receive these multicasts; ensure end-host ability before using this command.
Step 6	**ip irdp holdtime** *seconds* **Example:** `Device(config-if)#ip irdp holdtime 1000`	(Optional) Sets the IRDP period for which advertisements are valid. The default is three times the **maxadvertinterval** value. It must be greater than **maxadvertinterval** and cannot be greater than 9000 seconds. If you change the **maxadvertinterval** value, this value also changes.
Step 7	**ip irdp maxadvertinterval** *seconds* **Example:** `Device(config-if)#ip irdp maxadvertinterval 650`	(Optional) Sets the IRDP maximum interval between advertisements. The default is 600 seconds.
Step 8	**ip irdp minadvertinterval** *seconds* **Example:** `Device(config-if)#ip irdp minadvertinterval 500`	(Optional) Sets the IRDP minimum interval between advertisements. The default is 0.75 times the **maxadvertinterval**. If you change the **maxadvertinterval**, this value changes to the new default (0.75 of **maxadvertinterval**).
Step 9	**ip irdp preference** *number* **Example:** `Device(config-if)#ip irdp preference 2`	(Optional) Sets a device IRDP preference level. The allowed range is –231 to 231. The default is 0. A higher value increases the router preference level.
Step 10	**ip irdp address** *address* [*number*] **Example:** `Device(config-if)#ip irdp address 10.1.10.10`	(Optional) Specifies an IRDP address and preference to proxy-advertise.

	Command or Action	Purpose
Step 11	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 12	**show ip irdp** **Example:** `Device#show ip irdp`	Verifies settings by displaying IRDP values.
Step 13	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Broadcast Packet Handling

Perform the tasks in these sections to enable these schemes:

- Enabling Directed Broadcast-to-Physical Broadcast Translation
- Forwarding UDP Broadcast Packets and Protocols
- Establishing an IP Broadcast Address
- Flooding IP Broadcasts

Enabling Directed Broadcast-to-Physical Broadcast Translation

By default, IP directed broadcasts are dropped; they are not forwarded. Dropping IP-directed broadcasts makes routers less susceptible to denial-of-service attacks.

You can enable forwarding of IP-directed broadcasts on an interface where the broadcast becomes a physical (MAC-layer) broadcast. Only those protocols configured by using the **ip forward-protocol** global configuration command are forwarded.

You can specify an access list to control which broadcasts are forwarded. When an access list is specified, only those IP packets permitted by the access list are eligible to be translated from directed broadcasts to physical broadcasts. For more information on access lists, see the “Configuring ACLs" chapter in the *Security Configuration Guide*.

Note The **ip network-broadcast** command must be configured at the ingress interface before configuring the **ip directed-broadcast** command at the egress interface. This ensures that the IP-directed broadcasts work correctly and prevents an outage from occurring after an upgrade.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the interface to configure.
Step 4	**ip network-broadcast** **Example:** `Device(config-if)#ip network-broadcast`	Enables the ingress interface to receive and accept the network-prefix-directed broadcast packets.
Step 5	**exit** **Example:** `Device(config-if)#exit`	Returns to global configuration mode.
Step 6	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the interface to configure.
Step 7	**ip directed-broadcast** [*access-list-number*] **Example:** `Device(config-if)#ip directed-broadcast 103`	Enables directed broadcast-to-physical broadcast translation on the interface. You can include an access list to control which broadcasts are forwarded. When an access list, only IP packets permitted by the access list can be translated.
Step 8	**exit** **Example:** `Device(config-if)#exit`	Returns to global configuration mode.

	Command or Action	Purpose
Step 9	**ip forward-protocol** {**udp** [*port*] | **nd** | **sdns**} **Example:** `Device(config)#ip forward-protocol nd`	Specifies which protocols and ports the router forwards when forwarding broadcast packets. • **udp**—Forward UPD datagrams. port: (Optional) Destination port that controls which UDP services are forwarded. • **nd**—Forward ND datagrams. • **sdns**—Forward SDNS datagrams
Step 10	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 11	**show ip interface** [*interface-id*] **Example:** `Device#show ip interface`	Verifies the configuration on the interface or all interfaces
Step 12	**show running-config** **Example:** `Device#show running-config`	Verifies your entries.
Step 13	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Forwarding UDP Broadcast Packets and Protocols

If you do not specify any UDP ports when you configure the forwarding of UDP broadcasts, you are configuring the router to act as a BOOTP forwarding agent. BOOTP packets carry DHCP information.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
	`Device>enable`	
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 4	**ip helper-address** *address* **Example:** `Device(config-if)#ip helper address 10.1.10.1`	Enables forwarding and specifies the destination address for forwarding UDP broadcast packets, including BOOTP.
Step 5	**exit** **Example:** `Device(config-if)#exit`	Returns to global configuration mode.
Step 6	**ip forward-protocol** {**udp** [*port*] | **nd** | **sdns**} **Example:** `Device(config)#ip forward-protocol sdns`	Specifies which protocols the router forwards when forwarding broadcast packets.
Step 7	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 8	**show ip interface** [*interface-id*] **Example:** `Device#show ip interface gigabitethernet 1/1`	Verifies the configuration on the interface or all interfaces.
Step 9	**show running-config** **Example:** `Device#show running-config`	Verifies your entries.

	Command or Action	Purpose
Step 10	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Establishing an IP Broadcast Address

The most popular IP broadcast address (and the default) is an address consisting of all ones (255.255.255.255). However, the switch can be configured to generate any form of IP broadcast address.

To establish an IP broadcast address, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the interface to configure.
Step 4	**ip broadcast-address** *ip-address* **Example:** `Device(config-if)#ip broadcast-address 128.1.255.255`	Enters a broadcast address different from the default, for example 128.1.255.255.
Step 5	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show ip interface** [*interface-id*] **Example:** `Device#show ip interface`	Verifies the broadcast address on the interface or all interfaces.
Step 7	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Flooding IP Broadcasts

To configure IP broadcasts flooding, perform this procedure.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**ip forward-protocol spanning-tree** **Example:** `Device(config)#ip forward-protocol spanning-tree`	Uses the bridging spanning-tree database to flood UDP datagrams.
Step 4	**ip forward-protocol turbo-flood** **Example:** `Device(config)#ip forward-protocol turbo-flood`	Uses the spanning-tree database to speed up flooding of UDP datagrams.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)#end`	
Step 6	**show running-config** **Example:** `Device#show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

How to Configure IP Unicast Routing

The following sections provide configuration information about IP unicast routing.

Enabling IP Unicast Routing

By default, the device is in Layer 2 switching mode and IP routing is disabled. To use the Layer 3 capabilities of the device, you must enable IP routing.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**ip routing** **Example:** `Device(config)#ip routing`	Enables IP routing.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device#show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to Do Next

You can now set up parameters for the selected routing protocols as described in these sections:

- RIP
- OSPF,
- EIGRP
- Unicast Reverse Path Forwarding
- Protocol-Independent Features (optional)

Configuration Example for Enabling IP Routing

This example shows how to enable IP routing:

```
Device#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)#ip routing
Device(config-router)#end
```

Monitoring and Maintaining IP Addressing

When the contents of a particular cache, table, or database have become or are suspected to be invalid, you can remove all its contents by using the **clear** privileged EXEC commands. The Table lists the commands for clearing contents.

Table 38: Commands to Clear Caches, Tables, and Databases

Command	Purpose
clear arp-cache	Clears the IP ARP cache and the fast-switching cache.
clear host {*name* | ***}**	Removes one or all entries from the hostname and the address cache.
clear ip route {**network** [*mask*] | ***}**	Removes one or more routes from the IP routing table.

You can display specific statistics, such as the contents of IP routing tables, caches, and databases; the reachability of nodes; and the routing path that packets are taking through the network. The Table lists the privileged EXEC commands for displaying IP statistics.

Table 39: Commands to Display Caches, Tables, and Databases

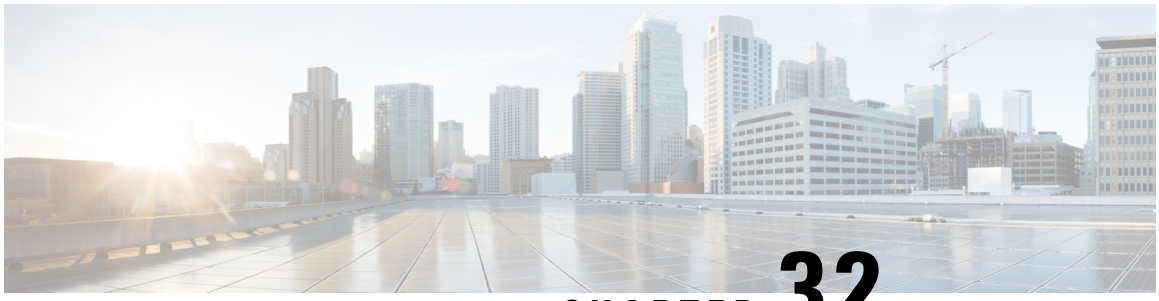
Command	Purpose
show arp	Displays the entries in the ARP table.
show hosts	Displays the default domain name, style of lookup service, name server hosts, and the cached list of hostnames and addresses.
show ip aliases	Displays IP addresses mapped to TCP ports (aliases).
show ip arp	Displays the IP ARP cache.
show ip interface [*interface-id*]	Displays the IP status of interfaces.
show ip irdp	Displays IRDP values.
show ip masks *address*	Displays the masks used for network addresses and the number of subnets using each mask.
show ip redirects	Displays the address of a default gateway.
show ip route [*address* [*mask*]] | [*protocol*]	Displays the current state of the routing table.
show ip route summary	Displays the current state of the routing table in summary form.

Monitoring and Maintaining the IP Network

You can remove all contents of a particular cache, table, or database. You can also display specific statistics.

Table 40: Command to Clear IP Routes or Display Route Status

Command	Purpose
show ip route summary	Displays the current state of the routing table in summary form.



CHAPTER 32

Configuring IPv6 Unicast Routing

Information About IPv6 Unicast Routing

This chapter describes how to configure IPv6 unicast routing on the switch.

Note To use all IPv6 features in this chapter, the switch or active switch must be running the Network Advantage license. Switches running the Network Essentials license support IPv6 static routing and RIP for IPv6. Switches running the Network Advantage license support OSPF and EIGRP for IPv6.

Understanding IPv6

IPv4 users can move to IPv6 and receive services such as end-to-end security, quality of service (QoS), and globally unique addresses. The IPv6 address space reduces the need for private addresses and Network Address Translation (NAT) processing by border routers at network edges.

For information about how Cisco Systems implements IPv6, go to Networking Software (IOS & NX-OS)

For information about IPv6 and other features in this chapter

- See the *Cisco IOS IPv6 Configuration Library*.
- Use the Search field on Cisco.com to locate the Cisco IOS software documentation. For example, if you want information about static routes, you can enter *Implementing Static Routes for IPv6* in the search field to learn about static routes.

Static Routes for IPv6

Static routes are manually configured and define an explicit route between two networking devices. Static routes are useful for smaller networks with only one path to an outside network or to provide security for certain types of traffic in a larger network.

Configuring Static Routing for IPv6 (CLI)

For configuring static routes for IPv6, see the *Configuring Static Routing for IPv6* section.

For more information about static routes, see the "Implementing Static Routes for IPv6" chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

Path MTU Discovery for IPv6 Unicast

The switch supports advertising the system maximum transmission unit (MTU) to IPv6 nodes and path MTU discovery. Path MTU discovery allows a host to dynamically discover and adjust to differences in the MTU size of every link along a given data path. In IPv6, if a link along the path is not large enough to accommodate the packet size, the source of the packet handles the fragmentation.

ICMPv6

The Internet Control Message Protocol (ICMP) in IPv6 generates error messages, such as ICMP destination unreachable messages, to report errors during processing and other diagnostic functions. In IPv6, ICMP packets are also used in the neighbor discovery protocol and path MTU discovery.

Neighbor Discovery

The switch supports NDP for IPv6, a protocol running on top of ICMPv6, and static neighbor entries for IPv6 stations that do not support NDP. The IPv6 neighbor discovery process uses ICMP messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), to verify the reachability of the neighbor, and to keep track of neighboring routers.

The switch supports ICMPv6 redirect for routes with mask lengths less than 64 bits. ICMP redirect is not supported for host routes or for summarized routes with mask lengths greater than 64 bits.

Neighbor discovery throttling ensures that the switch CPU is not unnecessarily burdened while it is in the process of obtaining the next hop forwarding information to route an IPv6 packet. The switch drops any additional IPv6 packets whose next hop is the same neighbor that the switch is actively trying to resolve. This drop avoids further load on the CPU.

Default Router Preference

The switch supports IPv6 default router preference (DRP), an extension in router advertisement messages. DRP improves the ability of a host to select an appropriate router, especially when the host is multihomed and the routers are on different links. The switch does not support the Route Information Option in RFC 4191.

An IPv6 host maintains a default router list from which it selects a router for traffic to offlink destinations. The selected router for a destination is then cached in the destination cache. NDP for IPv6 specifies that routers that are reachable or probably reachable are preferred over routers whose reachability is unknown or suspect. For reachable or probably reachable routers, NDP can either select the same router every time or cycle through the router list. By using DRP, you can configure an IPv6 host to prefer one router over another, provided both are reachable or probably reachable.

For configuring DRP for IPv6, see the *Configuring Default Router Preference* section.

For more information about DRP for IPv6, see the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

Destination Guard

The IPv6 Destination Guard feature works with IPv6 neighbor discovery to ensure that the device performs address resolution only for those addresses that are known to be active on the link. It relies on the address

glean functionality to populate all destinations active on the link into the binding table and then blocks resolutions before they happen when the destination is not found in the binding table.

For more information, see *Cisco IOS IPv6 Configuration Library* on Cisco.com.

MTU Path Discovery

IPv6 MTU Path Discovery allows a host to dynamically discover and adjust to differences in the maximum transmission unit (MTU) size of every link along a given data path.

As in IPv4, path MTU discovery in IPv6 allows a host to dynamically discover and adjust to differences in the MTU size of every link along a given data path. In IPv6, however, fragmentation is handled by the source of a packet when the path MTU of one link along a given data path is not large enough to accommodate the size of the packets. Having IPv6 hosts handle packet fragmentation saves IPv6 device processing resources and helps IPv6 networks run more efficiently.

For more information, see *Cisco IOS IPv6 Configuration Library* on Cisco.com.

Policy-Based Routing for IPv6

Policy-based routing (PBR) gives you a flexible means of routing packets by allowing you to configure a defined policy for traffic flows, which lessens reliance on routes that are derived from routing protocols. Therefore, PBR gives you more control over routing by extending and complementing the existing mechanisms that are provided by routing protocols. PBR allows you to set the IPv6 precedence. For a simple policy, you can use any one of these tasks; for a complex policy, you can use all of them. It also allows you to specify a path for certain traffic, such as priority traffic over a high-cost link.

PBR for IPv6 may be applied to both forwarded and originated IPv6 packets. For forwarded packets, PBR for IPv6 will be implemented as an IPv6 input interface feature, supported in the following forwarding paths:

- Process
- Cisco Express Forwarding (formerly known as CEF)
- Distributed Cisco Express Forwarding

Policies can be based on the IPv6 address, port numbers, protocols, or packet size.

PBR allows you to perform the following tasks:

- Classify traffic based on extended access list criteria. Access lists, then, establish the match criteria.
- Set IPv6 precedence bits, giving the network the ability to enable differentiated classes of service.
- Route packets to specific traffic-engineered paths; you might need to route them to allow a specific quality of service (QoS) through the network.

PBR allows you to classify and mark packets at the edge of the network. PBR marks a packet by setting precedence value. The precedence value can be used directly by devices in the network core to apply the appropriate QoS to a packet, which keeps packet classification at your network edge.

For enabling PBR for IPv6, see the *Enabling Local PBR for IPv6* section.

For enabling IPv6 PBR for an interface, see the *Enabling IPv6 PBR on an Interface* section.

Unsupported IPv6 Unicast Routing Features

The switch does not support these IPv6 features:

- IPv6 packets that are destined to site-local addresses.
- Tunneling protocols, such as IPv4-to-IPv6 or IPv6-to-IPv4.
- The switch as a tunnel endpoint supporting IPv4-to-IPv6 or IPv6-to-IPv4 tunneling protocols.
- IPv6 Web Cache Communication Protocol (WCCP).

IPv6 Feature Limitations

Because IPv6 is implemented in switch hardware, some limitations occurs due to the IPv6 compressed addresses in the hardware memory. This hardware limitation result in some loss of functionality and limits some features. For example, the switch cannot apply QoS classification on source-routed IPv6 packets in hardware.

Default IPv6 Configuration

Table 41: Default IPv6 Configuration

Feature	Default Setting
IPv6 routing	Disabled globally and on all interfaces
Cisco Express Forwarding for IPv6 or distributed Cisco Express Forwarding for IPv6	Disabled (IPv4 Cisco Express Forwarding and distributed Cisco Express Forwarding are enabled by default) **Note** When IPv6 routing is enabled, Cisco Express Forwarding for IPv6 and distributed Cisco Express Forwarding for IPv6 are automatically enabled.
IPv6 addresses	None configured

How to Configure IPv6 Unicast Routing

The following sections show the various configuration options available for IPv6 Unicast Routing

Configuring IPv6 Addressing and Enabling IPv6 Routing

This section describes how to assign IPv6 addresses to individual Layer 3 interfaces and to globally forward IPv6 traffic on the switch.

Note IPv6 routing is not enabled by default and needs to be enabled using the **ipv6 unicast-routing** command.

Before configuring IPv6 on the switch, consider these guidelines:

- Not all features that are discussed in this chapter are supported by the switch. See the Unsupported IPv6 Unicast Routing Features.

- In the **ipv6 address** interface configuration command, you must enter the *ipv6-address* and *ipv6-prefix* variables with the address that is specified in hexadecimal using 16-bit values between colons. The *prefix-length* variable (preceded by a slash [/]) is a decimal value that shows how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address).

To forward IPv6 traffic on an interface, you must configure a global IPv6 address on that interface. Configuring an IPv6 address on an interface automatically configures a link-local address and activates IPv6 for the interface. The configured interface automatically joins these required multicast groups for that link:

- solicited-node multicast group FF02:0:0:0:0:1:ff00::/104 for each unicast address assigned to the interface (this address is used in the neighbor discovery process.)
- all-nodes link-local multicast group FF02::1
- all-routers link-local multicast group FF02::2

To remove an IPv6 address from an interface, use the **no ipv6 address** *ipv6-prefix/prefix length* **eui-64** or **no ipv6 address** *ipv6-address* **link-local** interface configuration command. To remove all manually configured IPv6 addresses from an interface, use the **no ipv6 address** interface configuration command without arguments. To disable IPv6 processing on an interface that has not been explicitly configured with an IPv6 address, use the **no ipv6 enable** interface configuration command. To globally disable IPv6 routing, use the **no ipv6 unicast-routing** global configuration command.

For more information about configuring IPv6 routing, see the "Implementing Addressing and Basic Connectivity for IPv6" chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

To assign an IPv6 address to a Layer 3 interface and enable IPv6 routing, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode after the switch reloads.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure. The interface can be a physical interface, a switch virtual interface (SVI), or a Layer 3 EtherChannel.

	Command or Action	Purpose
Step 4	**no switchport** **Example:** `Device(config-if)# no switchport`	Removes the interface from Layer 2 configuration mode (if it is a physical interface).
Step 5	Use one of the following: • **ipv6 address** *ipv6-prefix/prefix length* **eui-64** • **ipv6 address** *ipv6-address/prefix length* • **ipv6 address** *ipv6-address* **link-local** • **ipv6 enable** • **ipv6 address** *WORD* • **ipv6 address** *autoconfig* • **ipv6 address** *dhcp* **Example:** `Device(config-if)# ipv6 address 2001:0DB8:c18:1::/64 eui 64` `Device(config-if)# ipv6 address 2001:0DB8:c18:1::/64` `Device(config-if)# ipv6 address 2001:0DB8:c18:1:: link-local` `Device(config-if)# ipv6 enable`	• Specifies a global IPv6 address with an extended unique identifier (EUI) in the low-order 64 bits of the IPv6 address. Specify only the network prefix; the last 64 bits are automatically computed from the switch MAC address. This enables IPv6 processing on the interface. • Manually configures an IPv6 address on the interface. • Specifies a link-local address on the interface to be used instead of the link-local address that is automatically configured when IPv6 is enabled on the interface. This command enables IPv6 processing on the interface. • Automatically configures an IPv6 link-local address on the interface, and enables the interface for IPv6 processing. The link-local address can only be used to communicate with nodes on the same link.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 7	**ip routing** **Example:** `Device(config)# ip routing`	Enables IP routing on the switch.
Step 8	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables forwarding of IPv6 unicast data packets.

	Command or Action	Purpose
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 10	**show ipv6 interface** *interface-id* **Example:** `Device# show ipv6 interface gigabitethernet 1/1`	Verifies your entries.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring IPv4 and IPv6 Protocol Stacks

Beginning in privileged EXEC mode, follow these steps to configure a Layer 3 interface to support both IPv4 and IPv6 and to enable IPv6 routing.

Note To disable IPv6 processing on an interface that has not been configured with an IPv6 address, use the **no ipv6 enable** command in interface configuration mode.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip routing** **Example:** `Device(config)# ip routing`	Enables routing on the switch.

	Command or Action	Purpose
Step 4	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables forwarding of IPv6 data packets on the switch.
Step 5	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 6	**no switchport** **Example:** `Device(config-if)# no switchport`	Removes the interface from Layer 2 configuration mode (if it is a physical interface).
Step 7	**ip address** *ip-address mask* [**secondary**] **Example:** `Device(config-if)# ip address 10.1.2.3 255.255.255`	Specifies a primary or secondary IPv4 address for the interface.
Step 8	Use one of the following: • **ipv6 address** *ipv6-prefix/prefix length* **eui-64** • **ipv6 address** *ipv6-address/prefix length* • **ipv6 address** *ipv6-address* **link-local** • **ipv6 enable** • **ipv6 address***WORD* • **ipv6 address***autoconfig* • **ipv6 address***dhcp*	• Specifies a global IPv6 address. Specify only the network prefix; the last 64 bits are automatically computed from the switch MAC address. • Specifies a link-local address on the interface to be used instead of the automatically configured link-local address when IPv6 is enabled on the interface. • Automatically configures an IPv6 link-local address on the interface, and enables the interface for IPv6 processing. The link-local address can only be used to communicate with nodes on the same link. **Note** To remove all manually configured IPv6 addresses from an interface, use the **no ipv6 address** interface configuration command without arguments.
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 10	Use one of the following: • **show interface** *interface-id*	Verifies your entries.

	Command or Action	Purpose
	• **show ip interface** *interface-id* • **show ipv6 interface** *interface-id*	
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Default Router Preference

Router advertisement messages are sent with the default router preference (DRP) configured by the **ipv6 nd router-preference** interface configuration command. If no DRP is configured, RAs are sent with a medium preference.

A DRP is useful when two routers on a link might provide equivalent, but not equal-cost routing, and policy might dictate that hosts should prefer one of the routers.

For more information about configuring DRP for IPv6, see the "Implementing IPv6 Addresses and Basic Connectivity" chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

Beginning in privileged EXEC mode, follow these steps to configure a DRP for a router on an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode and identifies the Layer 3 interface on which you want to specify the DRP.
Step 4	**ipv6 nd router-preference** {**high** | **medium** | **low**} **Example:** `Device(config-if)# ipv6 nd router-preference medium`	Specifies a DRP for the router on the switch interface.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
Step 6	**show ipv6 interface** **Example:** `Device# show ipv6 interface`	Verifies the configuration.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring IPv6 ICMP Rate Limiting

ICMP rate limiting is enabled by default with a default interval between error messages of 100 milliseconds and a bucket size (maximum number of tokens to be stored in a bucket) of 10.

To change the ICMP rate-limiting parameters, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 icmp error-interval** *interval* [*bucketsize*] **Example:** `Device(config)# ipv6 icmp error-interval 50 20`	Configures the interval and bucket size for IPv6 ICMP error messages: • *interval*—The interval (in milliseconds) between tokens being added to the bucket. The range is from 0 to 2147483647 milliseconds. • *bucketsize*—(Optional) The maximum number of tokens stored in the bucket. The range is from 1 to 200.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 5	**show ipv6 interface** [*interface-id*] **Example:** `Device# show ipv6 interface gigabitethernet 1/1`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Cisco Express Forwarding and distributed Cisco Express Forwarding for IPv6

Cisco Express Forwarding is a Layer 3 IP switching technology to improve network performance. Cisco Express Forwarding implements an advanced IP look-up and forwarding algorithm to deliver maximum Layer 3 switching performance. It is less CPU-intensive than fast-switching route-caching, allowing more CPU processing power to be dedicated to packet forwarding. IPv4 Cisco Express Forwarding and distributed Cisco Express Forwarding are enabled by default. IPv6 Cisco Express Forwarding and distributed Cisco Express Forwarding are disabled by default, but automatically enabled when you configure IPv6 routing.

IPv6 Cisco Express Forwarding and distributed Cisco Express Forwarding are automatically disabled when IPv6 routing is unconfigured. IPv6 Cisco Express Forwarding and distributed Cisco Express Forwarding cannot disabled through configuration. You can verify the IPv6 state by entering the **show ipv6 cef** command in privileged EXEC mode.

To route IPv6 unicast packets, you must first globally configure forwarding of IPv6 unicast packets by using the **ipv6 unicast-routing** global configuration command, and you must configure an IPv6 address and IPv6 processing on an interface by using the **ipv6 address** command in interface configuration mode.

For more information about configuring Cisco Express Forwarding and distributed Cisco Express Forwarding, see *Cisco IOS IPv6 Configuration Library* on Cisco.com.

Configuring Static Routing for IPv6

For more information about configuring static IPv6 routing, see the “Implementing Static Routes for IPv6” chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

To configure static IPv6 routing, perform this procedure:

Before you begin

You must enable routing by using the **ip routing** global configuration command, enable the forwarding of IPv6 packets by using the **ipv6 unicast-routing** command in global configuration mode, and enable IPv6 on at least one Layer 3 interface by configuring an IPv6 address on the interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route** *ipv6-prefix/prefix length* {*ipv6-address* | *interface-id* [*ipv6-address*]} [*administrative distance*] **Example:** `Device(config)# ipv6 route 2001:0DB8::/32 gigabitethernet 1/1 130`	Configures a static IPv6 route. • *ipv6-prefix*—The IPv6 network that is the destination of the static route. It can also be a hostname when static host routes are configured. • */prefix length*—The length of the IPv6 prefix. A decimal value that shows how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash mark must precede the decimal value. • *ipv6-address*—The IPv6 address of the next hop that can be used to reach the specified network. The IPv6 address of the next hop need not be directly connected; recursion is done to find the IPv6 address of the directly connected next hop. The address must be in the form that is documented in RFC 2373, specified in hexadecimal using 16-bit values between colons. • *interface-id*—Specifies direct static routes from point-to-point and broadcast interfaces. With point-to-point interfaces, there is no need to specify the IPv6 address of the next hop. With broadcast interfaces, you should always specify the IPv6 address of the next hop, or ensure that the specified prefix is assigned to the link, specifying a link-local address as the next hop. You can optionally specify the IPv6 address of the next hop to which packets are sent. **Note**

	Command or Action	Purpose
		You must specify an *interface-id* when using a link-local address as the next hop (the link-local next hop must also be an adjacent router). • *administrative distance*—(Optional) An administrative distance. The range is 1 to 254; the default value is 1, which gives static routes precedence over any other type of route except connected routes. To configure a floating static route, use an administrative distance greater than that of the dynamic routing protocol.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	Use one of the following: • **show ipv6 static** [*ipv6-address* | *ipv6-prefix/prefix length*] [**interface** *interface-id*] [**detail**]][**recursive**] [**detail**] • **show ipv6 route static** [*updated*] **Example:** `Device# show ipv6 static 2001:0DB8::/32 interface gigabitethernet 1/1` or `Device# show ipv6 route static`	Verifies your entries by displaying the contents of the IPv6 routing table. • **interface** *interface-id*—(Optional) Displays only those static routes with the specified interface as an egress interface. • **recursive**—(Optional) Displays only recursive static routes. The **recursive** keyword is mutually exclusive with the **interface** keyword, but it can be used with or without the IPv6 prefix included in the command syntax. • **detail**—(Optional) Displays this additional information: • For valid recursive routes, the output path set, and maximum resolution depth. • For invalid routes, the reason why the route is not valid.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Enabling IPv6 PBR on an Interface

To enable PBR for IPv6, you must create a route map that specifies the packet match criteria and desired policy-route action. Then you associate the route map on the required interface. All packets arriving on the specified interface that match the match clauses will be subject to PBR.

In PBR, the **set vrf** command decouples the virtual routing and forwarding (VRF) instance and interface association and allows the selection of a VRF based on access control list (ACL)-based classification using existing PBR or route-map configurations. It provides a single router with multiple routing tables and the ability to select routes based on ACL classification. The router classifies packets based on ACL, selects a routing table, looks up the destination address, and then routes the packet.

To enable PBR for IPv6, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map rip-to-ospf permit`	Defines the conditions for redistributing routes from one routing protocol into another, or enables policy routing, and enters route-map configuration mode.
Step 4	Do one of the following: • **match length** *minimum-length maximum-length* • **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*} **Example:** `Device(config-route-map)# match length 3 200` **Example:** `Device(config-route-map)# match ipv6 address marketing`	
Step 5	Do one of the following: • **set ipv6 next-hop** *global-ipv6-address* [*global-ipv6-address...*]	

	Command or Action	Purpose
	• **set ipv6 default next-hop** *global-ipv6-address* [*global-ipv6-address...*] **Example:** `Device(config-route-map)# set ipv6 next-hop 2001:DB8:2003:1::95` **Example:** `Device(config-route-map)# set ipv6 default next-hop 2001:DB8:2003:1::95`	
Step 6	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and returns to global configuration mode.
Step 7	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 8	**ipv6 policy route-map** *route-map-name* **Example:** `Device(config-if)# ipv6 policy-route-map interactive`	Identifies a route map to use for IPv6 PBR on an interface.
Step 9	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Enabling Local PBR for IPv6

Packets that are generated by the device are not normally policy routed. Perform this task to enable local IPv6 policy-based routing (PBR) for such packets, indicating which route maps the device should use.

To enable Local PBR for IPv6, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ipv6 local policy route-map** *route-map-name* **Example:** `Device(config)# ipv6 local policy route-map pbr-src-90`	Configures IPv6 PBR for packets that are generated by the device.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Displaying IPv6

For complete syntax and usage information on these commands, see the Cisco IOS command reference publications.

Table 42: Command for Monitoring IPv6

Command	Purpose
show ipv6 access-list	Displays a summary of access lists.
show ipv6 cef	Displays Cisco Express Forwarding for IPv6.
show ipv6 interface *interface-id*	Displays IPv6 interface status and configuration.
show ipv6 mtu	Displays IPv6 MTU per destination cache.
show ipv6 neighbors	Displays IPv6 neighbor cache entries.
show ipv6 prefix-list	Displays a list of IPv6 prefix lists.
show ipv6 protocols	Displays a list of IPv6 routing protocols on the switch.
show ipv6 rip	Displays IPv6 RIP routing protocol status.
show ipv6 route	Displays IPv6 route table entries.
show ipv6 static	Displays IPv6 static routes.
show ipv6 traffic	Displays IPv6 traffic statistics.

Configuration Examples for IPv6 Unicast Routing

The following sections show the various configuration examples available for IPv6 Unicast Routing

Example: Configuring IPv4 and IPv6 Protocol Stacks

This example shows how to enable IPv4 and IPv6 routing on an interface.

```
Device> enable
Device# configure terminal
Device(config)# ip routing
Device(config)# ipv6 unicast-routing
Devoce(config)# interface gigabitethernet1/1
Device(config-if)# no switchport
Device(config-if)# ip address 192.168.99.1 255.255.255.0
Device(config-if)# ipv6 address 2001:0DB8:c18:1::/64 eui 64
Device(config-if)# end
```

Example: Configuring Default Router Preference

This example shows how to configure a DRP of *high* for the router on an interface.

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet1/1
Device(config-if)# ipv6 nd router-preference high
Device(config-if)# end
```

Example: Configuring IPv6 ICMP Rate Limiting

This example shows how to configure an IPv6 ICMP error message interval of 50 milliseconds and a bucket size of 20 tokens.

```
Device> enable
Device# configure terminal
Device(config)#ipv6 icmp error-interval 50 20
```

Example: Configuring Static Routing for IPv6

This example shows how to configure a floating static route to an interface with an administrative distance of 130:

```
Device> enable
Device# configure terminal
Device(config)# ipv6 route 2001:0DB8::/32 gigabitethernet 1/1 130
```

Example: Enabling PBR on an Interface

In the following example, a route map that is named pbr-dest-1 is created and configured, specifying packet match criteria and desired policy-route action. PBR is then enabled on GigabitEthernet interface 1/1.

```
Device> enable
Device# configure terminal
Device(config)# ipv6 access-list match-dest-1
Device(config)# permit ipv6 any 2001:DB8:2001:1760::/32
Device(config)# route-map pbr-dest-1 permit 10
Device(config)# match ipv6 address match-dest-1
Device(config)# set interface GigabitEthernet 1/1
```

```
Device(config)# interface GigabitEthernet1/1
Device(config-if)# ipv6 policy-route-map interactive
```

Example: Enabling Local PBR for IPv6

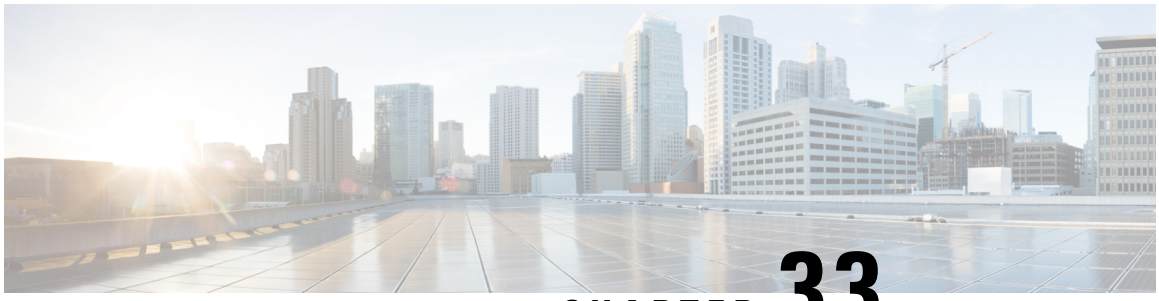
In the following example, packets with a destination IPv6 address that match the IPv6 address range allowed by access list pbr-src-90 are sent to the device at IPv6 address 2001:DB8:2003:1::95:

```
Device> enable
Device# configure terminal
Device(config)# ipv6 access-list src-90
Device(config)# permit ipv6 host 2001:DB8:2003::90 2001:DB8:2001:1000::/64
Device(config)# route-map pbr-src-90 permit 10
Device(config)# match ipv6 address src-90
Device(config)# set ipv6 next-hop 2001:DB8:2003:1::95
Device(config)# ipv6 local policy route-map pbr-src-90
```

Example: Displaying IPv6

This is an example of the output from the **show ipv6 interface** command:

```
Device> enable
Device# show ipv6 interface
Vlan1 is up, line protocol is up
  IPv6 is enabled, link-local address is GE80::20B:46FF:FE2F:D940
  Global unicast address(es):
    3FFE:C000:0:1:20B:46FF:FE2F:D940, subnet is 3FFE:C000:0:1::/64 [EUI]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF2F:D940
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
<output truncated>
```



CHAPTER 33

Configuring RIP

Information About RIP

The Routing Information Protocol (RIP) is an interior gateway protocol (IGP) created for use in small, homogeneous networks. It is a distance-vector routing protocol that uses broadcast User Datagram Protocol (UDP) data packets to exchange routing information. The protocol is documented in RFC 1058. You can find detailed information about RIP in *IP Routing Fundamentals,* published by Cisco Press.

Note RIP is supported in the Network Essentials feature set.

Using RIP, the switch sends routing information updates (advertisements) every 30 seconds. If a router does not receive an update from another router for 180 seconds or more, it marks the routes served by that router as unusable. If there is still no update after 240 seconds, the router removes all routing table entries for the non-updating router.

RIP uses hop counts to rate the value of different routes. The hop count is the number of routers that can be traversed in a route. A directly connected network has a hop count of zero; a network with a hop count of 16 is unreachable. This small range (0 to 15) makes RIP unsuitable for large networks.

If the router has a default network path, RIP advertises a route that links the router to the pseudonetwork 0.0.0.0. The 0.0.0.0 network does not exist; it is treated by RIP as a network to implement the default routing feature. The switch advertises the default network if a default was learned by RIP or if the router has a gateway of last resort and RIP is configured with a default metric. RIP sends updates to the interfaces in specified networks. If an interface's network is not specified, it is not advertised in any RIP update.

RIP for IPv6

Routing Information Protocol (RIP) for IPv6 is a distance-vector protocol that uses hop count as a routing metric. It includes support for IPv6 addresses and prefixes and the all-RIP-routers multicast group address FF02::9 as the destination address for RIP update messages.

For configuring RIP for IPv6, see the *Configuring RIP for IPv6* section.

For more information about RIP for IPv6, see the "Implementing RIP for IPv6" chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

Summary Addresses and Split Horizon

Routers connected to broadcast-type IP networks and using distance-vector routing protocols normally use the split-horizon mechanism to reduce the possibility of routing loops. Split horizon blocks information about routes from being advertised by a router on any interface from which that information originated. This feature usually optimizes communication among multiple routers, especially when links are broken.

How to Configure Routing Information Protocol

The following sections provide configurational information about RIP.

Default RIP Configuration

Table 43: Default RIP Configuration

Feature	Default Setting
Auto summary	Enabled.
Default-information originate	Disabled.
Default metric	Built-in; automatic metric translations.
IP RIP authentication key-chain	No authentication. Authentication mode: clear text.
IP RIP triggered	Disabled
IP split horizon	Varies with media.
Neighbor	None defined.
Network	None specified.
Offset list	Disabled.
Output delay	0 milliseconds.
Timers basic	• Update: 30 seconds. • Invalid: 180 seconds. • Hold-down: 180 seconds. • Flush: 240 seconds.
Validate-update-source	Enabled.

Feature	Default Setting
Version	Receives RIP Version 1 and 2 packets; sends Version 1 packets.

Configuring Basic RIP Parameters

To configure RIP, you enable RIP routing for a network and optionally configure other parameters. On the switch, RIP configuration commands are ignored until you configure the network number.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip routing** **Example:** `Device(config)# ip routing`	Enables IP routing. (Required only if IP routing is disabled.)
Step 4	**router rip** **Example:** `Device(config)# router rip`	Enables a RIP routing process, and enter router configuration mode.
Step 5	**network** *network number* **Example:** `Device(config-router)# network 12.0.0.0`	Associates a network with a RIP routing process. You can specify multiple **network** commands. RIP routing updates are sent and received through interfaces only on these networks. **Note** You must configure a network number for the RIP commands to take effect.
Step 6	**neighbor** *ip-address* **Example:**	(Optional) Defines a neighboring router with which to exchange routing information. This step allows routing updates from RIP

	Command or Action	Purpose
	`Device(config-router)# neighbor 10.2.5.1`	(normally a broadcast protocol) to reach nonbroadcast networks.
Step 7	**offset-list** [*access-list number* | *name*] {**in** | **out**} *offset* [*type number*] **Example:** `Device(config-router)# offset-list 103 in 10`	(Optional) Applies an offset list to routing metrics to increase incoming and outgoing metrics to routes learned through RIP. You can limit the offset list with an access list or an interface.
Step 8	**timers basic** *update invalid holddown flush* **Example:** `Device(config-router)# timers basic 45 360 400 300`	(Optional) Adjusts routing protocol timers. Valid ranges for all timers are 0 to 4294967295 seconds. • *update*—The time between sending routing updates. The default is 30 seconds. • *invalid*—The timer after which a route is declared invalid. The default is 180 seconds. • *holddown*—The time before a route is removed from the routing table. The default is 180 seconds. • *flush*—The amount of time for which routing updates are postponed. The default is 240 seconds.
Step 9	**version** {**1** | **2**} **Example:** `Device(config-router)# version 2`	(Optional) Configures the switch to receive and send only RIP Version 1 or RIP Version 2 packets. By default, the switch receives Version 1 and 2 but sends only Version 1. You can also use the interface commands **ip rip** {**send** | **receive**} **version 1** | **2** | **1 2**} to control what versions are used for sending and receiving on interfaces.
Step 10	**no auto summary** **Example:** `Device(config-router)# no auto summary`	(Optional) Disables automatic summarization. By default, the switch summarizes subprefixes when crossing classful network boundaries. Disable summarization (RIP Version 2 only) to advertise subnet and host routing information to classful network boundaries.
Step 11	**output-delay** *delay* **Example:** `Device(config-router)# output-delay 8`	(Optional) Adds interpacket delay for RIP updates sent. By default, packets in a multiple-packet RIP update have no delay added between packets. If you are sending packets to a lower-speed device, you can add

	Command or Action	Purpose
		an interpacket delay in the range of 8 to 50 milliseconds.
Step 12	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.
Step 13	**show ip protocols** **Example:** `Device# show ip protocols`	Verifies your entries.
Step 14	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring RIP Authentication

RIP Version 1 does not support authentication. If you are sending and receiving RIP Version 2 packets, you can enable RIP authentication on an interface. The key chain specifies the set of keys that can be used on the interface. If a key chain is not configured, no authentication is performed, not even the default.

The switch supports two modes of authentication on interfaces for which RIP authentication is enabled: plain text and MD5. The default is plain text.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:**	Enters interface configuration mode, and specifies the interface to configure.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet 1/1`	
Step 4	**ip rip authentication key-chain** *name-of-chain* **Example:** `Device(config-if)# ip rip authentication key-chain trees`	Enables RIP authentication.
Step 5	**ip rip authentication mode** {**text** | **md5**} **Example:** `Device(config-if)# ip rip authentication mode md5`	Configures the interface to use plain text authentication (the default) or MD5 digest authentication.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring RIP for IPv6

For more information about configuring RIP routing for IPv6, see the "Implementing RIP for IPv6" chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com,

To configure RIP routing for IPv6, perform this procedure:

Before you begin

Before configuring the switch to run IPv6 RIP, you must enable routing by using the **ip routing** command in global configuration mode, enable the forwarding of IPv6 packets by using the **ipv6 unicast-routing** command in global configuration mode, and enable IPv6 on any Layer 3 interfaces on which IPv6 RIP is to be enabled.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router rip** *name* **Example:** `Device(config)# ipv6 router rip cisco`	Configures an IPv6 RIP routing process, and enters router configuration mode for the process.
Step 4	**maximum-paths** *number-paths* **Example:** `Device(config-router)# maximum-paths 6`	(Optional) Define the maximum number of equal-cost routes that IPv6 RIP can support. The range is from 1 to 32, and the default is 16 routes.
Step 5	**exit** **Example:** `Device(config-router)# exit`	Returns to global configuration mode.
Step 6	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 7	**ipv6 rip** *name* **enable** **Example:** `Device(config-if)# ipv6 rip cisco enable`	Enables the specified IPv6 RIP routing process on the interface.
Step 8	**ipv6 rip** *name* **default-information** {**only** | **originate**} **Example:** `Device(config-if)# ipv6 rip cisco default-information only`	(Optional) Originates the IPv6 default route (::/0) into the RIP routing process updates sent from the specified interface. **Note** To avoid routing loops after the IPv6 default route (::/0) is originated from any interface, the routing process ignores all default routes received on any interface. • **only**—Select to originate the default route, but suppress all other routes in the updates sent on this interface.

	Command or Action	Purpose
		• **originate**—Select to originate the default route in addition to all other routes in the updates sent on this interface.
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 10	Use one of the following: • **show ipv6 rip** [*name*] [**interface** *interface-id*] [**database**] [**next-hops**] • **show ipv6 rip** **Example:** `Device# show ipv6 rip cisco interface gigabitethernet 1/1` or `Device# show ipv6 rip`	• Displays information about current IPv6 RIP processes. • Displays the current contents of the IPv6 routing table.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Summary Addresses and Split Horizon

Note In general, disabling split horizon is not recommended unless you are certain that your application requires it to properly advertise routes.

If you want to configure an interface running RIP to advertise a summarized local IP address pool on a network access server for dial-up clients, use the **ip summary-address rip** interface configuration command.

Note If split horizon is enabled, neither autosummary nor interface IP summary addresses are advertised.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 4	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 10.1.1.10 255.255.255.0`	Configures the IP address and IP subnet.
Step 5	**ip summary-address rip ip address** *ip-network mask* **Example:** `Device(config-if)# ip summary-address rip ip address 10.1.1.30 255.255.255.0`	Configures the IP address to be summarized and the IP network mask.
Step 6	**no ip split horizon** **Example:** `Device(config-if)# no ip split horizon`	Disables split horizon on the interface.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show ip interface** *interface-id* **Example:** `Device# show ip interface gigabitethernet 1/1`	Verifies your entries.
Step 9	**copy running-config startup-config** **Example:**	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`Device# copy running-config startup-config`	

Configuring Split Horizon

Routers connected to broadcast-type IP networks and using distance-vector routing protocols normally use the split-horizon mechanism to reduce the possibility of routing loops. Split horizon blocks information about routes from being advertised by a router on any interface from which that information originated. This feature can optimize communication among multiple routers, especially when links are broken.

Note In general, we do not recommend disabling split horizon unless you are certain that your application requires it to properly advertise routes.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the interface to configure.
Step 4	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 10.1.1.10 255.255.255.0`	Configures the IP address and IP subnet.
Step 5	**no ip split-horizon** **Example:**	Disables split horizon on the interface.

	Command or Action	Purpose
	`Device(config-if)# no ip split-horizon`	
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show ip interface** *interface-id* **Example:** `Device# show ip interface gigabitethernet 1/1`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuration Examples for Routing Information Protocol

The following sections provide configuration examples for RIP.

Configuration Example for Summary Addresses and Split Horizon

In this example, the major net is 10.0.0.0. The summary address 10.2.0.0 overrides the autosummary address of 10.0.0.0 so that 10.2.0.0 is advertised out interface Gigabit Ethernet port 2, and 10.0.0.0 is not advertised. In the example, if the interface is still in Layer 2 mode (the default), you must enter a **no switchport** interface configuration command before entering the **ip address** interface configuration command.

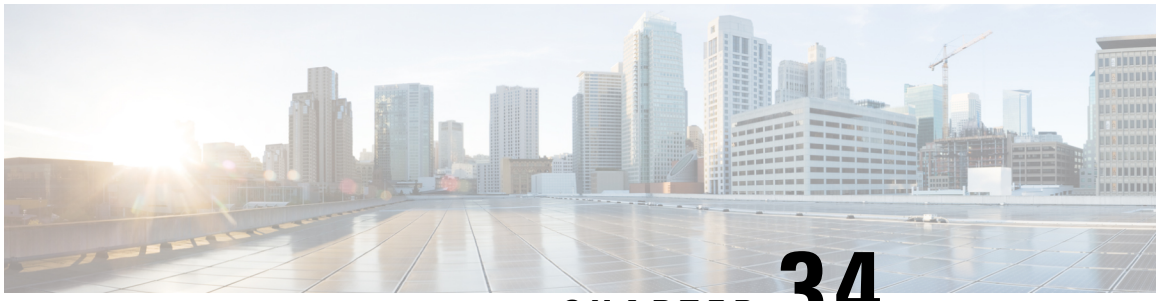
Note If split horizon is enabled, neither autosummary nor interface summary addresses (those configured with the **ip summary-address rip** router configuration command) are advertised.

```
Device(config)# router rip
Device(config-router)# interface gigabitethernet1/1
Device(config-if)# ip address 10.1.5.1 255.255.255.0
Device(config-if)# ip summary-address rip 10.2.0.0 255.255.0.0
Device(config-if)# no ip split-horizon
Device(config-if)# exit
Device(config)# router rip
Device(config-router)# network 10.0.0.0
Device(config-router)# neighbor 2.2.2.2 peer-group mygroup
Device(config-router)# end
```

Example: Configuring RIP for IPv6

This example shows how to enable the RIP routing process *cisco* with a maximum of eight equal-cost routes and to enable it on an interface:

```
Device> enable
Devce# configure terminal
Device(config)# ipv6 router rip cisco
Device(config-router)# maximum-paths 8
Device(config)# exit
Device(config)# interface gigabitethernet1/1
Device(config-if)# ipv6 rip cisco enable
```



CHAPTER 34

Configuring OSPF

Information About OSPF

OSPF is an Interior Gateway Protocol (IGP) designed expressly for IP networks, supporting IP subnetting and tagging of externally derived routing information. OSPF also allows packet authentication and uses IP multicast when sending and receiving packets. The Cisco implementation supports RFC 1253, OSPF management information base (MIB).

The Cisco implementation conforms to the OSPF Version 2 specifications with these key features:

- Definition of stub areas is supported.
- Routes learned through any IP routing protocol can be redistributed into another IP routing protocol. At the intradomain level, this means that OSPF can import routes learned through EIGRP and RIP. OSPF routes can also be exported into RIP.
- Plain text and MD5 authentication among neighboring routers within an area is supported.
- Configurable routing interface parameters include interface output cost, retransmission interval, interface transmit delay, router priority, router dead and hello intervals, and authentication key.
- Virtual links are supported.
- Not-so-stubby-areas (NSSAs) per RFC 1587are supported.

OSPF typically requires coordination among many internal routers, area border routers (ABRs) connected to multiple areas, and autonomous system boundary routers (ASBRs). The minimum configuration would use all default parameter values, no authentication, and interfaces assigned to areas. If you customize your environment, you must ensure coordinated configuration of all routers.

OSPF for IPv6

The switch supports Open Shortest Path First (OSPF) for IPv6, a link-state protocol for IP.

Note The Network Essentials license allows configuration of 1000 routes only. To configure more than 1000 routes, Network Advantage license is required.

For configuring OSPF for IPv6, see the *Configuring OSPF for IPv6* section.

For more information, see *Cisco IOS IPv6 Configuration Library* on Cisco.com.

OSPF Area Parameters

You can optionally configure several OSPF area parameters. These parameters include authentication for password-based protection against unauthorized access to an area, stub areas, and not-so-stubby-areas (NSSAs). Stub areas are areas into which information on external routes is not sent. Instead, the area border router (ABR) generates a default external route into the stub area for destinations outside the autonomous system (AS). An NSSA does not flood all LSAs from the core into the area, but can import AS external routes within the area by redistribution.

Route summarization is the consolidation of advertised addresses into a single summary route to be advertised by other areas. If network numbers are contiguous, you can use the **area range** router configuration command to configure the ABR to advertise a summary route that covers all networks in the range.

Other OSPF Parameters

You can optionally configure other OSPF parameters in router configuration mode.

- Route summarization: When redistributing routes from other protocols. Each route is advertised individually in an external LSA. To help decrease the size of the OSPF link state database, you can use the **summary-address** router configuration command to advertise a single router for all the redistributed routes included in a specified network address and mask.
- Virtual links: In OSPF, all areas must be connected to a backbone area. You can establish a virtual link in case of a backbone-continuity break by configuring two Area Border Routers as endpoints of a virtual link. Configuration information includes the identity of the other virtual endpoint (the other ABR) and the nonbackbone link that the two routers have in common (the transit area). Virtual links cannot be configured through a stub area.
- Default route: When you specifically configure redistribution of routes into an OSPF routing domain, the route automatically becomes an autonomous system boundary router (ASBR). You can force the ASBR to generate a default route into the OSPF routing domain.
- Domain Name Server (DNS) names for use in all OSPF **show** privileged EXEC command displays makes it easier to identify a router than displaying it by router ID or neighbor ID.
- Default Metrics: OSPF calculates the OSPF metric for an interface according to the bandwidth of the interface. The metric is calculated as *ref-bw* divided by bandwidth, where *ref* is 10 by default, and bandwidth (*bw*) is specified by the **bandwidth** interface configuration command. For multiple links with high bandwidth, you can specify a larger number to differentiate the cost on those links.
- Administrative distance is a rating of the trustworthiness of a routing information source, an integer between 0 and 255, with a higher value meaning a lower trust rating. An administrative distance of 255 means the routing information source cannot be trusted at all and should be ignored. OSPF uses three different administrative distances: routes within an area (interarea), routes to another area (interarea),

and routes from another routing domain learned through redistribution (external). You can change any of the distance values.

- Passive interfaces: Because interfaces between two devices on an Ethernet represent only one network segment, to prevent OSPF from sending hello packets for the sending interface, you must configure the sending device to be a passive interface. Both devices can identify each other through the hello packet for the receiving interface.
- Route calculation timers: You can configure the delay time between when OSPF receives a topology change and when it starts the shortest path first (SPF) calculation and the hold time between two SPF calculations.
- Log neighbor changes: You can configure the router to send a syslog message when an OSPF neighbor state changes, providing a high-level view of changes in the router.

LSA Group Pacing

The OSPF LSA group pacing feature allows the router to group OSPF LSAs and pace the refreshing, check-summing, and aging functions for more efficient router use. This feature is enabled by default with a 4-minute default pacing interval, and you will not usually need to modify this parameter. The optimum group pacing interval is inversely proportional to the number of LSAs the router is refreshing, check-summing, and aging. For example, if you have approximately 10,000 LSAs in the database, decreasing the pacing interval would benefit you. If you have a very small database (40 to 100 LSAs), increasing the pacing interval to 10 to 20 minutes might benefit you slightly.

Loopback Interfaces

OSPF uses the highest IP address configured on the interfaces as its router ID. If this interface is down or removed, the OSPF process must recalculate a new router ID and resend all its routing information out its interfaces. If a loopback interface is configured with an IP address, OSPF uses this IP address as its router ID, even if other interfaces have higher IP addresses. Because loopback interfaces never fail, this provides greater stability. OSPF automatically prefers a loopback interface over other interfaces, and it chooses the highest IP address among all loopback interfaces.

How to Configure OSPF

Default OSPF Configuration

Table 44: Default OSPF Configuration

Feature	Default Setting
Interface parameters	Retransmit interval: 5 seconds. Transmit delay: 1 second. Priority: 1. Hello interval: 10 seconds. Dead interval: 4 times the hello interval. No authentication. No password specified. MD5 authentication disabled.
Area	Authentication type: 0 (no authentication). Default cost: 1. Range: Disabled. Stub: No stub area defined. NSSA: No NSSA area defined.
Auto cost	100 Mb/s.
Default-information originate	Disabled. When enabled, the default metric setting is 10, and the external route type default is Type 2.
Default metric	Built-in, automatic metric translation, as appropriate for each routing protocol.
Distance OSPF	dist1 (all routes within an area): 110. dist2 (all routes from one area to another): 110. and dist3 (routes from other routing domains): 110.
OSPF database filter	Disabled. All outgoing link-state advertisements (LSAs) are flooded to the interface.
IP OSPF name lookup	Disabled.
Log adjacency changes	Enabled.
Neighbor	None specified.
Neighbor database filter	Disabled. All outgoing LSAs are flooded to the neighbor.

Feature	Default Setting
Network area	Disabled.
Router ID	No OSPF routing process defined.
Summary address	Disabled.
Timers LSA group pacing	240 seconds.
Timers shortest path first (spf)	spf delay: 50 miliseconds; spf-holdtime: 200 miliseconds.
Virtual link	No area ID or router ID defined. Hello interval: 10 seconds. Retransmit interval: 5 seconds. Transmit delay: 1 second. Dead interval: 40 seconds. Authentication key: no key predefined. Message-digest key (MD5): no key predefined.

Configuring Basic OSPF Parameters

To enable OSPF, create an OSPF routing process, specify the range of IP addresses to associate with the routing process, and assign area IDs to be associated with that range.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**router ospf** process-id **Example:** `Device(config)#router ospf 15`	Enables OSPF routing, and enter router configuration mode. The process ID is an internally used identification parameter that is locally assigned and can be any positive integer. Each OSPF routing process has a unique value. **Note**

	Command or Action	Purpose
		OSPF for Routed Access supports only one OSPFv2 and one OSPFv3 instance with a maximum number of 1000 dynamically learned routes.
Step 4	**network** address wildcard-mask **area** area-id **Example:** `Device(config-router)#network 10.1.1.1 255.240.0.0 area 20`	Define an interface on which OSPF runs and the area ID for that interface. You can use the wildcard-mask to use a single command to define one or more multiple interfaces to be associated with a specific OSPF area. The area ID can be a decimal value or an IP address.
Step 5	**end** **Example:** `Device(config-router)#end`	Returns to privileged EXEC mode.
Step 6	**show ip protocols** **Example:** `Device#show ip protocols`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring OSPF for IPv6

For more information about configuring OSPF routing for IPv6, see the "Implementing OSPF for IPv6" chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

To configure OSPF routing for IPv6, perform this procedure:

Before you begin

You can customize OSPF for IPv6 for your network. However, the defaults for OSPF in IPv6 are set to meet the requirements of most customers and features.

Follow these guidelines:

- Be careful when changing the defaults for IPv6 commands. Changing the defaults might adversely affect OSPF for the IPv6 network.
- Before you enable IPv6 OSPF on an interface, you must enable routing by using the **ip routing** command in global configuration mode, enable the forwarding of IPv6 packets by using the **ipv6 unicast-routing**

command in global configuration mode, and enable IPv6 on Layer 3 interfaces on which you are enabling IPv6 OSPF.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 21`	Enables OSPF router configuration mode for the process. The process ID is the number assigned administratively when enabling the OSPF for IPv6 routing process. It is locally assigned and can be a positive integer from 1 to 65535.
Step 4	**area** *area-id* **range** {*ipv6-prefix/prefix length*} [**advertise** | **not-advertise**] [**cost** *cost*] **Example:** `Device(config)# area .3 range 2001:0DB8::/32 not-advertise`	(Optional) Consolidates and summarizes routes at an area boundary. • *area-id*—Identifier of the area about which routes are to be summarized. It can be specified as either a decimal value or as an IPv6 prefix. • *ipv6-prefix/prefix length*—The destination IPv6 network and a decimal value that shows how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash mark (/) must precede the decimal value. • **advertise**—(Optional) Sets the address range status to advertise and generate a Type 3 summary link-state advertisement (LSA). • **not-advertise**—(Optional) Sets the address range status to DoNotAdvertise. The Type 3 summary LSA is suppressed, and component networks remain hidden from other networks. • **cost** *cost*—(Optional) Sets the metric or cost for this summary route, which is used during OSPF SPF calculation to

	Command or Action	Purpose
		determine the shortest paths to the destination. The value can be 0 to 16777215.
Step 5	**maximum paths** *number-paths* **Example:** `Device(config)# maximum paths 16`	(Optional) Defines the maximum number of equal-cost routes to the same destination that IPv6 OSPF should enter in the routing table. The range is from 1 to 32, and the default is 16 paths.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 7	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 8	**ipv6 ospf** *process-id* **area** *area-id* [**instance** *instance-id*] **Example:** `Device(config-if)# ipv6 ospf 21 area .3`	Enables OSPF for IPv6 on the interface. • **instance** *instance-id*—(Optional) Instance identifier.
Step 9	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 10	Use one of the following: • **show ipv6 ospf** [*process-id*] [*area-id*] **interface** [*interface-id*] • **show ipv6 ospf** [*process-id*] [*area-id*] **Example:** `Device# show ipv6 ospf 21 interface gigabitethernet1/1` or `Device# show ipv6 ospf 21`	• Displays information about OSPF interfaces. • Displays general information about OSPF routing processes.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring OSPF Interfaces

You can use the **ip ospf** interface configuration commands to modify interface-specific OSPF parameters. You are not required to modify any of these parameters, but some interface parameters (hello interval, dead interval, and authentication key) must be consistent across all routers in an attached network. If you modify these parameters, be sure all routers in the network have compatible values.

Note The **ip ospf** interface configuration commands are all optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** interface-id **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 4	**ip ospf cost** cost **Example:** `Device(config-if)#ip ospf cost 8`	(Optional) Explicitly specifies the cost of sending a packet on the interface.
Step 5	**ip ospf retransmit-interval** seconds **Example:** `Device(config-if)#ip ospf transmit-interval 10`	(Optional) Specifies the number of seconds between link state advertisement transmissions. The range is 1 to 65535 seconds. The default is 5 seconds.
Step 6	**ip ospf transmit-delay** seconds **Example:** `Device(config-if)#ip ospf transmit-delay 2`	(Optional) Sets the estimated number of seconds to wait before sending a link state update packet. The range is 1 to 65535 seconds. The default is 1 second.

	Command or Action	Purpose
Step 7	**ip ospf priority** number **Example:** `Device(config-if)#ip ospf priority 5`	(Optional) Sets priority to help find the OSPF designated router for a network. The range is from 0 to 255. The default is 1.
Step 8	**ip ospf hello-interval** seconds **Example:** `Device(config-if)#ip ospf hello-interval 12`	(Optional) Sets the number of seconds between hello packets sent on an OSPF interface. The value must be the same for all nodes on a network. The range is 1 to 65535 seconds. The default is 10 seconds.
Step 9	**ip ospf dead-interval** seconds **Example:** `Device(config-if)#ip ospf dead-interval 8`	(Optional) Sets the number of seconds after the last device hello packet was seen before its neighbors declare the OSPF router to be down. The value must be the same for all nodes on a network. The range is 1 to 65535 seconds. The default is 4 times the hello interval.
Step 10	**ip ospf authentication-key** key **Example:** `Device(config-if)#ip ospf authentication-key password`	(Optional) Assign a password to be used by neighboring OSPF routers. The password can be any string of keyboard-entered characters up to 8 bytes in length. All neighboring routers on the same network must have the same password to exchange OSPF information.
Step 11	**ip ospf message digest-key** keyid **md5** key **Example:** `Device(config-if)#ip ospf message digest-key 16 md5 your1pass`	(Optional) Enables MDS authentication. • *keyid*—An identifier from 1 to 255. • *key*—An alphanumeric password of up to 16 bytes.
Step 12	**ip ospf database-filter all out** **Example:** `Device(config-if)#ip ospf database-filter all out`	(Optional) Block flooding of OSPF LSA packets to the interface. By default, OSPF floods new LSAs over all interfaces in the same area, except the interface on which the LSA arrives.
Step 13	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 14	**show ip ospf interface** [*interface-name*] **Example:** `Device#show ip ospf interface`	Displays OSPF-related interface information.

	Command or Action	Purpose
Step 15	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring OSPF Area Parameters

Before you begin

Note The OSPF **area** router configuration commands are all optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**router ospf** process-id **Example:** `Device(config)#router ospf 109`	Enables OSPF routing, and enter router configuration mode.
Step 4	**area** *area-id* **authentication** **Example:** `Device(config-router)#area 1 authentication`	(Optional) Allow password-based protection against unauthorized access to the identified area. The identifier can be either a decimal value or an IP address.
Step 5	**area** *area-id* **authentication message-digest** **Example:** `Device(config-router)#area 1 authentication message-digest`	(Optional) Enables MD5 authentication on the area.

	Command or Action	Purpose
Step 6	**area** *area-id* **stub** [**no-summary**] **Example:** `Device(config-router)#area 1 stub`	(Optional) Define an area as a stub area. The **no-summary** keyword prevents an ABR from sending summary link advertisements into the stub area.
Step 7	**area** *area-id* **nssa** [**no-redistribution**] [**default-information-originate**] [**no-summary**] **Example:** `Device(config-router)#area 1 nssa default-information-originate`	(Optional) Defines an area as a not-so-stubby-area. Every router within the same area must agree that the area is NSSA. Select one of these keywords: • **no-redistribution**—Select when the router is an NSSA ABR and you want the **redistribute** command to import routes into normal areas, but not into the NSSA. • **default-information-originate**—Select on an ABR to allow importing type 7 LSAs into the NSSA. • **no-redistribution**—Select to not send summary LSAs into the NSSA.
Step 8	**area** *area-id* **range** *address mask* **Example:** `Device(config-router)#area 1 range 255.240.0.0`	(Optional) Specifies an address range for which a single route is advertised. Use this command only with area border routers.
Step 9	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 10	**show ip ospf** [*process-id*] **Example:** `Device#show ip ospf`	Displays information about the OSPF routing process in general or for a specific process ID to verify configuration.
Step 11	**show ip ospf** [*process-id* [*area-id*]] **database** **Example:** `Device#show ip osfp database`	Displays lists of information related to the OSPF database for a specific router.
Step 12	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Other OSPF Parameters

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**router ospf** *process-id* **Example:** `Device(config)#router ospf 10`	Enables OSPF routing, and enter router configuration mode.
Step 4	**summary-address** *address mask* **Example:** `Device(config)#summary-address 10.1.1.1 255.255.255.0`	(Optional) Specifies an address and IP subnet mask for redistributed routes so that only one summary route is advertised.
Step 5	**area** *area-id* **virtual-link** *router-id* [**hello-interval** *seconds*] [**retransmit-interval** *seconds*] [**trans**] [[**authentication-key** *key*] | **message-digest-key** *keyid* **md5** *key*]] **Example:** `Device(config)#area 2 virtual-link 192.168.255.1 hello-interval 5`	(Optional) Establishes a virtual link and set its parameters.
Step 6	**default-information originate** [**always**] [**metric** *metric-value*] [**metric-type** *type-value*] [**route-map** *map-name*] **Example:** `Device(config)#default-information originate metric 100 metric-type 1`	(Optional) Forces the ASBR to generate a default route into the OSPF routing domain. Parameters are all optional.
Step 7	**ip ospf name-lookup** **Example:** `Device(config)#ip ospf name-lookup`	(Optional) Configures DNS name lookup. The default is disabled.

	Command or Action	Purpose
Step 8	**ip auto-cost reference-bandwidth** *ref-bw* **Example:** `Device(config)#ip auto-cost reference-bandwidth 5`	(Optional) Specifies an address range for which a single route will be advertised. Use this command only with area border routers.
Step 9	**distance ospf** {[**inter-area** *dist1*] [**inter-area** *dist2*] [**external** *dist3*]} **Example:** `Device(config)#distance ospf inter-area 150`	(Optional) Changes the OSPF distance values. The default distance for each type of route is 110. The range is 1 to 255.
Step 10	**passive-interface** *type number* **Example:** `Device(config)#passive-interface gigabitethernet 1/1`	(Optional) Suppresses the sending of hello packets through the specified interface.
Step 11	**timers throttle spf** *spf-delay spf-holdtime spf-wait* **Example:** `Device(config)#timers throttle spf 200 100 100`	(Optional) Configures route calculation timers. • *spf-delay*—Delay between receiving a change to SPF calculation. The range is from 1 to 600000 miliseconds. • *spf-holdtime*—Delay between first and second SPF calculation. The range is from 1 to 600000 in milliseconds. • *spf-wait*—Maximum wait time in milliseconds for SPF calculations. The range is from 1 to 600000 in milliseconds.
Step 12	**ospf log-adj-changes** **Example:** `Device(config)#ospf log-adj-changes`	(Optional) Sends syslog message when a neighbor state changes.
Step 13	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 14	**show ip ospf** [*process-id* [*area-id*]] **database** **Example:** `Device#show ip ospf database`	Displays lists of information related to the OSPF database for a specific router.

	Command or Action	Purpose
Step 15	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Changing LSA Group Pacing

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**router ospf** *process-id* **Example:** `Device(config)#router ospf 25`	Enables OSPF routing, and enter router configuration mode.
Step 4	**timers lsa-group-pacing** *seconds* **Example:** `Device(config-router)#timers lsa-group-pacing 15`	Changes the group pacing of LSAs.
Step 5	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device#show running-config`	Verifies your entries.

	Command or Action	Purpose
Step 7	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Loopback Interface

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface loopback 0** **Example:** `Device(config)#interface loopback 0`	Creates a loopback interface, and enter interface configuration mode.
Step 4	**ip address** address mask **Example:** `Device(config-if)#ip address 10.1.1.5 255.255.240.0`	Assign an IP address to this interface.
Step 5	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 6	**show ip interface** **Example:** `Device#show ip interface`	Verifies your entries.

	Command or Action	Purpose
Step 7	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring OSPF

You can display specific statistics such as the contents of IP routing tables, caches, and databases.

Table 45: Show IP OSPF Statistics Commands

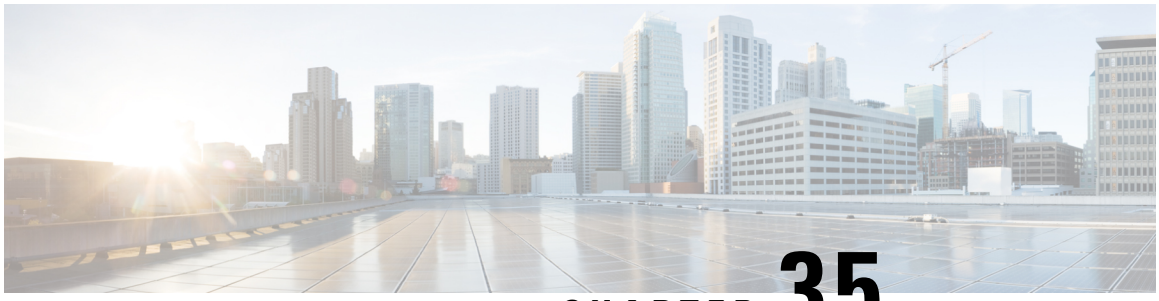
Command	Purpose
show ip ospf [*process-id*]	Displays general information about OSPF routing processes.
show ip ospf [*process-id*] **database** [**router**] [*link-state-id*] **show ip ospf** [*process-id*] **database** [**router**] [**self-originate**] **show ip ospf** [*process-id*] **database** [**router**] [**adv-router** [*ip-address*]] **show ip ospf** [*process-id*] **database** [**network**] [*link-state-id*] **show ip ospf** [*process-id*] **database** [**summary**] [*link-state-id*] **show ip ospf** [*process-id*] **database** [**asbr-summary**] [*link-state-id*] **show ip ospf** [*process-id*] **database** [**external**] [*link-state-id*] **show ip ospf** [*process-id area-id*] **database** [**database-summary**]	Displays lists of information related to the OSPF database.
show ip ospf border-routes	Displays the internal OSPF routing ABR and ASBR table entries.
show ip ospf interface [*interface-name*]	Displays OSPF-related interface information.
show ip ospf neighbor [*interface-name*] [*neighbor-id*] **detail**	Displays OSPF interface neighbor information.
show ip ospf virtual-links	Displays OSPF-related virtual links information.

Configuration Examples for OSPF

Example: Configuring Basic OSPF Parameters

This example shows how to configure an OSPF routing process and assign it a process number of 109:

```
Device(config)#router ospf 109
Device(config-router)#network 131.108.0.0 255.255.255.0 area 24
```



CHAPTER 35

Configuring OSPF Link-State Database Overload Protection

Information About OSPF Link-State Database Overload Protection

The OSPF Link-State Database Overload Protection feature allows you to limit the number of nonself-generated link-state advertisements (LSAs) for a given Open Shortest Path First (OSPF) process or OSPFv3 process. Excessive LSAs generated by other devices in the OSPF domain can substantially drain the CPU and memory resources of the device.

The OSPF Link-State Database Overload Protection feature is applicable to OSPF, OSPFv2 and OSPFv3.

Benefits of Using OSPF Link-State Database Overload Protection

The OSPF Link-State Database Overload Protection feature provides a mechanism at the OSPF level to limit the number of nonself-generated LSAs for a given OSPF process. When other devices in the network have been misconfigured, they may generate a high volume of LSAs, for instance, to redistribute large numbers of prefixes. This protection mechanism prevents devices from receiving a large number of LSAs and therefore experiencing CPU and memory shortages.

Overview of OSPF Link-State Database Overload Protection

When the OSPF Link-State Database Overload Protection feature is enabled, the device keeps a count of the number of nonself-generated LSAs that it has received. When the configured threshold number of LSAs is reached, an error message is logged. When the configured maximum number of LSAs is exceeded, the device sends a notification. If the count of received LSAs is still higher than the configured maximum after one minute, the OSPF process takes down all adjacencies and clears the OSPF database. In this ignore state, all OSPF packets received on any interface that belong to this OSPF process are ignored and no OSPF packets are generated on any of these interfaces. The OSPF process remains in the ignore state for the time configured by the **ignore-time** keyword of the **max-lsa** command. Each time the OSPF process gets into an ignore state

a counter is incremented. If this counter exceeds the number of times configured by the **ignore-count** keyword, the OSPF process stays permanently in the same ignore state and manual intervention is required to get the OSPF process out of the ignore state. You can get the OSPF process out of the permanent ignore state by restarting the OSPF process. The ignore state counter is reset to 0 when the OSPF process remains in the normal state of operation for the amount of time that was specified by the **reset-time** keyword. If the **warning-only** keyword of the **max-lsa** command is configured, the OSPFprocess will send only a warning that the LSA maximum has been exceeded.

How to Configure OSPF Link-State Database Overload Protection

Limiting the Number of Non Self-Generated LSAs for an OSPF Process

To configure a limit for the number of non self-generated LSAs for an OPSF process, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospf***process-id* **Example:** `Device(config)# router ospf 1`	Enables OSPF routing. The *process-id* argument identifies the OSPF process.
Step 4	**router-id** *ip-address* **Example:** `Device(config-router)# router-id 10.0.0.1`	Specifies a fixed router ID for an OSPF process.
Step 5	**log-adjacency-changes** [**detail**] **Example:** `Device(config-router)# log-adjacency-changes`	Configures the device to send a syslog message when an OSPF neighbor goes up or down.
Step 6	**max-lsa** *maximum number* [*threshold-percentage*] [**warning-only**] [**ignore-time** *minutes*] [**ignore-count***count-number*] [**reset-time***minutes*] **Example:** `Device(config-router)# max-lsa 12000`	Limits the number of non self-generated LSAs that an OSPF routing process can keep in the OSPF link-state database (LSDB). • The default limit for the number of non self-generated LSAs is 50,000 LSAs. • The default value for the *threshold* argument is 75 percent.

	Command or Action	Purpose
		• The default value for the **ignore-time** argument is 5 minutes. • The default value for the **reset-time** argument is 10 minutes. • The default value for the **ignore-count** argument is 5 counts.
Step 7	**network** *ip-address wildcard-mask* **area** *area-id* **Example:** `Device(config-router)# network 209.165.201.1 255.255.255.255 area 0`	Defines the interfaces on which OSPF runs and defines the area ID for those interfaces.
Step 8	**end** **Example:** `Device(config-router)# end`	
Step 9	**show ip ospf** [*process-id area-id*] **database** [**database-summary**] **Example:** `Device# show ip ospf 2000 database database-summary`	Displays lists of information related to the OSPF database for a specific device. Use this command to verify the number of non self-generated LSAs on a device.

Limiting the Number of Non Self-Generated LSAs for an OSPFv3 Process

To configure a limit for the number of non self-generated LSAs for an OPSFv3 process, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3***process-id* **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 routing. The *process-id* argument identifies the OSPFv3 process.

	Command or Action	Purpose
Step 4	**router-id** *ip-address* **Example:** `Device(config-router)# router-id 10.0.0.1`	Specifies a fixed router ID for an OSPF process.
Step 5	**log-adjacency-changes** [**detail**] **Example:** `Device(config-router)# log-adjacency-changes`	Configures the device to send a syslog message when an OSPF neighbor goes up or down.
Step 6	**max-lsa** *maximum number* [*threshold-percentage*] [**warning-only**] [**ignore-time** *minutes*] [**ignore-count***count-number*] [**reset-time***minutes*] **Example:** `Device(config-router)# max-lsa 12000`	Limits the number of non self-generated LSAs that an OSPF routing process can keep in the OSPF link-state database (LSDB). • The default limit for the number of non self-generated LSAs is 50,000 LSAs. • The default value for the *threshold* argument is 75 percent. • The default value for the **ignore-time** argument is 5 minutes. • The default value for the **reset-time** argument is 10 minutes. • The default value for the **ignore-count** argument is 5 counts.
Step 7	**end** **Example:** `Device(config-router)# end`	
Step 8	**show ospfv3** [*process-id area-id*] **database** [**database-summary**] **Example:** `Device# show ospfv3 2000 database database-summary`	Displays lists of information related to the OSPF database for a specific device. Use this command to verify the number of non self-generated LSAs on a device.

Configuration Examples for OSPF Link-State Database Overload Protection

Example: Setting a Limit for LSA Generation

In the following example, the device is configured to not accept any more non self-generated LSAs once a maximum of 14,000 has been exceeded:

```
Device(config)# router ospf 1
Device(config-router)# router-id 192.168.0.1
Device(config-router)# log-adjacency-changes
Device(config-router)# max-lsa 14000
Device(config-router)# area 33 nssa
Device(config-router)# network 192.168.0.10.0.0.0 area 1
Device(config-router)# network 192.168.5.10.0.0.0 area 1
Device(config-router)# network 192.168.2.10.0.0.0 area 0
```

In the following example, the device is configured to not accept any more non self-generated LSAs once a maximum of 12,000 has been exceeded for an OPSFv3 process:

```
Device> enable
Device# configure terminal
Device(config)# router ospfv3 1
Device(config-router)# router-id 10.0.0.1
Device(config-router)# log-adjacency-changes
Device(config-router)# max-lsa 12000
```

In the following example, the **show ip ospf** command is entered to confirm the configuration:

```
Device# show ip ospf 1
Routing Process "ospf1" with ID 192.168.0.1
Supports only single TOS(TOS0)routes
Supports opaque LSA
Supports Link-local Signaling(LLS)
Supports area transit capability
Maximum number of nonself-generated LSA allowed 14000
Threshold for warning message75%
Ignore-time 5minutes,reset-time 10minutes
Ignore-countallowed 5,currentignore-count 0
```

In the following example, the output is displayed when the **show ip ospf** command is entered when the device is in the ignore state:

```
Device# show ip ospf 1
Routing Process "ospf1" with ID 192.168.0.1
Supports only single TOS(TOS0)routes
Supports opaque LSA
Supports Link-local Signaling(LLS)
Supports area transit capability
Maximum number of nonself-generated LSA allowed 14000
Threshold for warning message 75%
Ignore-time 5minutes, reset-time 10minutes
Ignore-count allowed 5,current ignore-count 1
Ignoring all neighbors due to max-lsa limit, time remaining: 00:04:52
```

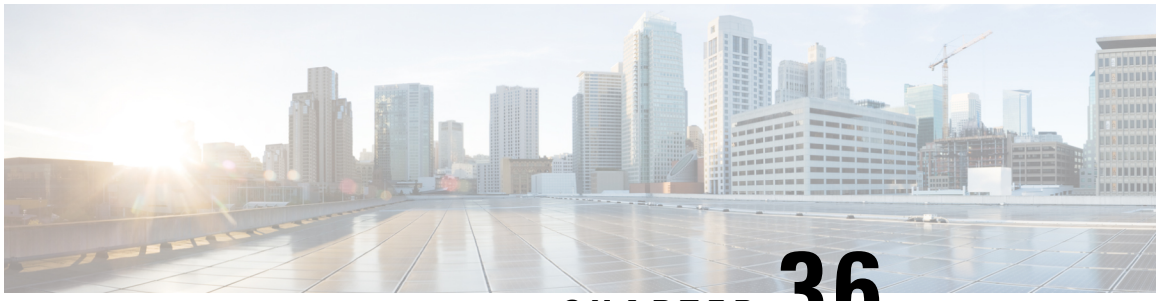
The following output is displayed when the **show ip ospf** command is entered after the device left the ignore state:

```
Device# show ip ospf 1
Routing Process "ospf 1" with ID 192.168.0.1
Supports only single TOS(TOS0) routes
Supports opaque LSA Supports Link-local Signaling (LLS)
Supports area transit capability
Maximum number of non self-generated LSA allowed 14000
Threshold for warning message 75%
```

```
Ignore-time 5 minutes, reset-time 10 minutes
Ignore-count allowed 5, current ignore-count 1- time remaining: 00:09:51
```

The following output is displayed when the **show ip ospf** command is entered for a device that is permanently in the ignore state:

```
Device# show ip ospf 1
Routing Process "ospf 1" with ID 192.168.0.1
Supports only single TOS(TOS0) routes
Supports opaque LSA Supports Link-local Signaling (LLS)
Supports area transit capability
Maximum number of non self-generated LSA allowed 14000
Threshold for warning message 75%
Ignore-time 5 minutes, reset-time 10 minutes
Ignore-count allowed 5, current ignore-count 6
Permanently ignoring all neighbors due to max-lsa limit
```



CHAPTER 36

Configuring OSPF Limit on Number of Redistributed Routes

Restrictions for OSPF Limit on Number of Redistributed Routes

OSPFv3 Limit on Number of Redistributed Routes is supported only for the IPv6 address family.

Prerequisites for OSPF Limit on Number of Redistributed Routes

You must have Open Shortest Path First (OSPF) configured in your network either along with another protocol, or another OSPF process for redistribution.

Information About OSPF Limit on Number of Redistributed Routes

OSPF supports a user-defined maximum number of prefixes (routes) that can be redistributed into OSPF from other protocols or other OSPF processes. Such a limit helps prevent the device from being flooded by too many redistributed routes.

For example, if a large number of IP routes are sent into OSPF for a network that allows redistribution of Border Gateway Protocol (BGP) into OSPF, the network can get severely flooded. Limiting the number of redistributed routes prevents this potential problem.

The command **redistribute maximum-prefix** *maximum[threshold]* is enabled with the default number of routes set at 10240 routes. The default number of routes is to protect the OSPF processes from being flooded with routes. You can still configure the number of routes using the **redistribute maximum-prefix** command.

The OSPF Limit on Number of Redistributed Routes feature is applicable to OSPF, OSPFv2 and OSPFv3.

How to Configure an OSPF Limit on the Number of Redistributed Routes

The following sections provide information on configuring an OSPF limit on the number of redistributed routes.

Note The following procedures are mutually exclusive, that is, you can either limit the number of redistributed routes, or request a warning about the number of routes redistributed into OSPF.

Limiting the Number of OSPF Redistributed Routes

This task describes how to limit the number of OSPF redistributed routes. If the number of redistributed routes reaches the maximum value configured, no more routes are redistributed.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospf** *process-id* **Example:** `Device(config)# router ospf 1`	Configures an OSPF routing process.
Step 4	**redistribute** *protocol* [*process-id*] [*as-number*] [**include-connected** {**level-1** | **level-1-2** | **level-2**} [**metric** *metric-value*] [**metric-type** *type-value*] [**nssa-only**] [**tag** *tag-value*] [**route-map** *map-tag*] **Example:** `Device(config-router-af)# redistribute eigrp 10`	Redistributes routes from one routing domain into another routing domain.
Step 5	**redistribute maximum-prefix** *maximum* [*threshold*] **Example:**	Sets a maximum number of IP prefixes that are allowed to be redistributed into OSPF. • The default value for the *maximum* argument is set at 10240 routes.

	Command or Action	Purpose
	`Device(config-router-af)# redistribute maximum-prefix 100 80`	• The *threshold* value defaults to 75 percent. **Note** If the **warning-only** keyword is configured in this command, no limit is enforced; a warning message is logged.
Step 6	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode.

Limiting the Number of OSPFv3 Redistributed Routes

This task describes how to limit the number of OSPFv3 redistributed routes. If the number of redistributed routes reaches the maximum value configured, no more routes are redistributed.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** *process-id* **Example:** `Device(config)# router ospfv3 1`	Configures an OSPFv3 routing process.
Step 4	**address-family ipv6** [**unicast**] **Example:** `Device(config-router)# address-family ipv6 unicast`	Enters IPv6 address family configuration mode.
Step 5	**redistribute** *protocol* [*process-id*] [*as-number*] [**include-connected** {**level-1** | **level-1-2** | **level-2**} [**metric** *metric-value*] [**metric-type** *type-value*] [**nssa-only**] [**tag** *tag-value*] [**route-map** *map-tag*] **Example:** `Device(config-router-af)# redistribute eigrp 10`	Redistributes routes from one routing domain into another routing domain.

	Command or Action	Purpose
Step 6	**redistribute maximum-prefix** *maximum* [*threshold*] **Example:** `Device(config-router-af)# redistribute maximum-prefix 100 80`	Sets a maximum number of IPv6 prefixes that are allowed to be redistributed into OSPFv3. • The default value for the *maximum* argument is set at 10240 routes. • The *threshold* value defaults to 75 percent. **Note** If the **warning-only** keyword is configured in this command, no limit is enforced; a warning message is logged.
Step 7	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits IPv6 address family configuration mode.
Step 8	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode.

Requesting a Warning Message About the Number of Routes Redistributed into OSPF

To request a warning message when the number of routes redistributed into OSPF exceeds the configuration limit, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospf** *process-id* **Example:** `Device(config)# router ospf 1`	Configures an OSPF routing process.

	Command or Action	Purpose
Step 4	**redistribute** *protocol* [*process-id*] [*as-number*] [**include-connected** {**level-1** | **level-1-2** | **level-2**} [**metric** *metric-value*] [**metric-type** *type-value*] [**nssa-only**] [**tag** *tag-value*] [**route-map** *map-tag*] **Example:** `Device(config-router-af)# redistribute eigrp 10`	Redistributes routes from one routing domain into another routing domain.
Step 5	**redistribute maximum-prefix** *maximum* [*threshold*] [**warning-only**] **Example:** `Device(config-router-af)# redistribute maximum-prefix 1000 80 warning-only`	Causes a warning message to be logged when the maximum number of IP prefixes have been redistributed to OSPFv3. • Because the **warning-only** keyword is included, no limit is imposed on the number of redistributed prefixes into OSPF. • The *threshold* value defaults to 75 percent. • This example causes two warnings: one at 80 percent of 1000 (800 routes redistributed) and another at 1000 routes redistributed
Step 6	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode.

Requesting a Warning Message About the Number of Routes Redistributed into OSPFv3

To request a warning message when the number of routes redistributed into OSPFv3 exceeds the configuration limit, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**router ospfv3** *process-id* **Example:** `Device(config)# router ospfv3 1`	Configures an OSPFv3 routing process.
Step 4	**address-family ipv6** [**unicast**] **Example:** `Device(config-router)# address-family ipv6 unicast`	Enters IPv6 address family configuration mode.
Step 5	**redistribute** *protocol* [*process-id*] [*as-number*] [**include-connected** {**level-1** | **level-1-2** | **level-2**} [**metric** *metric-value*] [**metric-type** *type-value*] [**nssa-only**] [**tag** *tag-value*] [**route-map** *map-tag*] **Example:** `Device(config-router-af)# redistribute eigrp 10`	Redistributes routes from one routing domain into another routing domain.
Step 6	**redistribute maximum-prefix** *maximum* [*threshold*] [**warning-only**] **Example:** `Device(config-router-af)# redistribute maximum-prefix 1000 80 warning-only`	Causes a warning message to be logged when the maximum number of IP prefixes have been redistributed to OSPFv3. • Because the **warning-only** keyword is included, no limit is imposed on the number of redistributed prefixes into OSPFv3. • The *threshold* value defaults to 75 percent. • This example causes two warnings: one at 80 percent of 1000 (800 routes redistributed) and another at 1000 routes redistributed
Step 7	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode.

Configuration Examples for OSPF Limit on Number of Redistributed Routes

The following sections provide configuration examples for OSPF Limit on Number of Redistributed Routes.

Example: OSPF Limit on Number of Redistributed Routes

This example shows how to set a maximum of 1200 prefixes that can be redistributed into the OSPF process 1. Prior to reaching the limit, when the number of prefixes that are redistributed reaches 80 percent of 1200 (960 prefixes), a warning message is logged. Another warning message is logged when the limit is reached and no more routes are redistributed.

```
Device> enable
Device# configure terminal
Device(config)# router ospf 1
Device(config-router-af)# redistribute static subnets
Device(config-router-af)# redistribute maximum-prefix 1200 80
```

This example shows how to set a maximum of 1200 prefixes that can be redistributed into the OSPFv3 process 1.

```
Device> enable
Device# configure terminal
Device(config)# router ospfv3 1
Device(config-router)# address-family ipv6
Device(config-router-af)# redistribute static subnets
Device(config-router-af)# redistribute maximum-prefix 1200 80
```

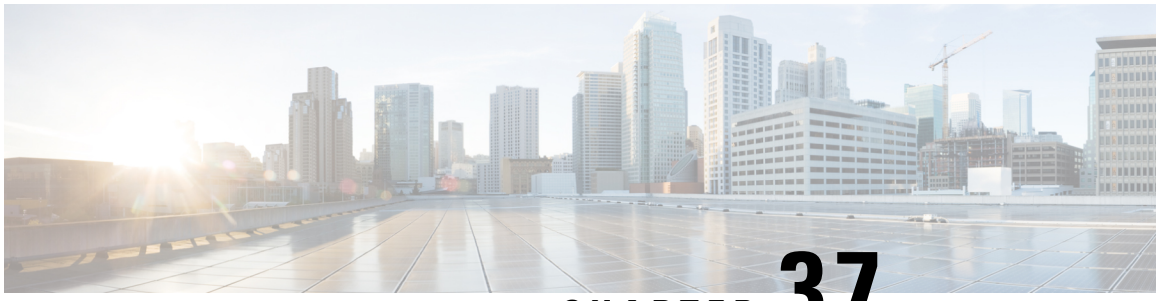
Example: Requesting a Warning Message About the Number of Redistributed Routes

This example shows how to enable two warning messages to be logged, the first if the number of prefixes that are redistributed reaches 85 percent of 600 (510 prefixes), and the second if the number of redistributed routes reaches 600. However, the number of redistributed routes is not limited.

```
Device> enable
Device# configure terminal
Device(config)# router ospf 11
Device(config-router-af)# redistribute eigrp 10 subnets
Device(config-router-af)# redistribute maximum-prefix 600 85 warning-only
```

This example shows how to enable two warnings to be logged for an OSSPv3 process.

```
Device> enable
Device# configure terminal
Device(config)# router ospfv3 11
Device(config-router)# address-family ipv6
Device(config-router-af)# redistribute eigrp 10 subnets
Device(config-router-af)# redistribute maximum-prefix 600 85 warning-only
```



CHAPTER 37

Configuring OSPF Local RIB

Prerequisite for OSPF Local RIB

Before configuring this feature, ensure the OSPF routing protocol is configured.

Restriction for OSPF Local RIB

This feature is available only for IP Version 4 networks.

Information About OSPF Local RIB

A router that is running OSPF maintains a local RIB in which it stores all routes to destinations that it has learned from its neighbors. At the end of each SPF, OSPF attempts to install the best (that is, the least-cost) routes to a destination present in the local RIB into the global IPv4 routing table. The global RIB will be updated only when routes are added, deleted, or changed. Routes in the local RIB and Forwarding Information Base (FIB) will not compute when intermediate results are computed during SPF, resulting in fewer dropped packets in some circumstances.

By default, the contents of the global RIB are used to compute inter-area summaries, NSSA translation, and forwarding addresses for type-5 and type-7 LSAs. Each of these functions can be configured to use the contents of the OSPF local RIB instead of the global RIB for their computation. Using the local RIB for the computation may be slightly faster in some circumstances, but because the local RIB has information for only a particular instance of OSPF, using it for the computation may yield incorrect results. Potential problems that may occur include routing loops and null routes. It is recommended that you not change the default values because they are conservative and preserve the current global RIB behavior.

By default, OSPF installs discard routes to null0 for any area range (internal) or summary-address (external) prefixes that it advertises to other routers. Installation of a discard route can prevent routing loops in cases where portions of a summary do not have a more specific route in the RIB. Normally, internal discard routes

are installed with an administrative distance of 110, while external discard routes have an administrative distance of 254.

There may be rare circumstances, however, when some other values are needed. For example, if one OSPF process installs a route that exactly matches an area range configured on another OSPF process, the internal discard routes for the second OSPF process could be given a higher (less desirable) administrative distance.

OSPF Local RIB Path Limit

The OSPF Local RIB Path Limit feature allows network administrators to control the number of paths OSPF installs in its Local RIB for a specific prefix.

How to Configure OSPF Local RIB

Although it is recommended to keep the default settings for the commands described in the following sections, it is optional to change the defaults settings.

Changing the Default Local RIB Criteria

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospf** *process-id* [**vrf** *vpn-name*] **Example:** `Device(config)# router ospf 23`	Configures an OSPFv2 routing process and enters router configuration mode.
Step 4	**local-rib-criteria** [**forwarding-address**] [**inter-area-summary**] [**nssa-translation**] **Example:** `Device(config-router)# local-rib-criteria forwarding-address`	Specifies that the OSPF local RIB will be used for route validation.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-router)# end`	
Step 6	**show ip ospf** *process-id* **rib** [**redistribution**] [*network-prefix*] [*network-mask*] [**detail**] **Example:** `Device# show ip ospf 23 rib`	Displays information for the OSPF local RIB or locally redistributed routes.

Changing the Administrative Distance for Discard Routes

It is recommended that you keep the default settings. However, you can follow the steps in this section to change the administrative distance for discard routes.

Procedure

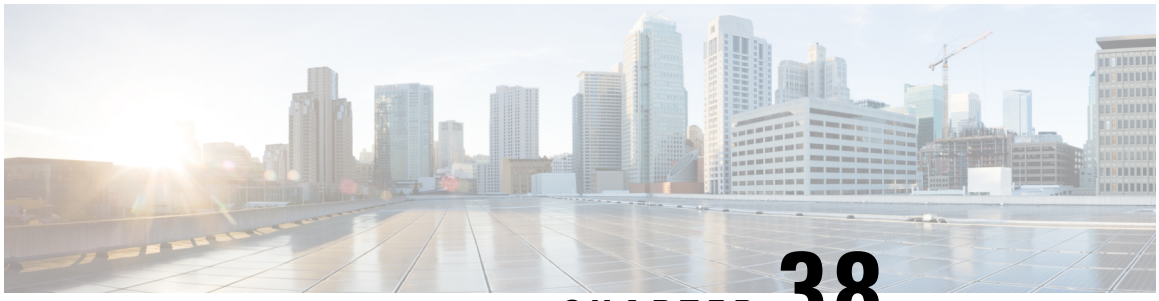
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospf** *process-id* [**vrf** *vpn-name*] **Example:** `Device(config)# router ospf 23`	Configures an OSPFv2 routing process and enters router configuration mode.
Step 4	**discard-route** [**external** [*distance*]] [**internal** [*distance*]] **Example:** `Device(config-router)# discard-route external 150`	Specifies the administrative distance to be used for internal and external discard routes. **Note** You can now specify the administrative distance for internal and external discard routes.
Step 5	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show ip route** [*ip-address* [*mask*] [**longer-prefixes**] | *protocol* [*process-id*] | **list** [*access-list-number* | *access-list-name*] | **static download**] **Example:** `Device# show ip route ospf 23`	Displays the current state of the routing table. **Note** Entering the **show ip route** command will verify the changed administrative distance values for external and internal discard routes.

Example

The sample output displayed for the **show ip route** command confirms that the administrative distance for the IP route 192.168.0.0/24 is 110.

```
Device# show ip route 192.168.0.0 255.255.255.0

Routing entry for 192.168.0.0/24

          Known via "ospf 1", distance 110, metric 0, type intra area

  Routing Descriptor Blocks:

  * directly connected, via Null0

       Route metric is 0, traffic share count is 1
```



CHAPTER 38

Configuring EIGRP

Information About EIGRP

Enhanced IGRP (EIGRP) is a Cisco proprietary enhanced version of the IGRP. EIGRP uses the same distance vector algorithm and distance information as IGRP; however, the convergence properties and the operating efficiency of EIGRP are improved.

The convergence technology employs an algorithm referred to as the Diffusing Update Algorithm (DUAL), which guarantees loop-free operation at every instant throughout a route computation and allows all devices that are involved in a topology change to synchronize at the same time. Routers that are not affected by topology changes are not involved in recomputations.

IP EIGRP provides increased network width. With RIP, the largest possible width of your network is 15 hops. Because the EIGRP metric is large enough to support thousands of hops, the only barrier to expanding the network is the transport-layer hop counter. EIGRP increments the transport control field only when an IP packet has traversed 15 routers and the next hop to the destination was learned through EIGRP. When a RIP route is used as the next hop to the destination, the transport control field is incremented as usual.

EIGRP IPv6

Switches support the Enhanced Interior Gateway Routing Protocol (EIGRP) for IPv6. It is configured on the interfaces on which it runs and does not require a global IPv6 address. Switches running Network Essentials only support EIGRPv6 stub routing.

Before running, an instance of EIGRP IPv6 requires an implicit or explicit router ID. An implicit router ID is derived from a local IPv6 address, so any IPv6 node always has an available router ID. However, EIGRP IPv6 might be running in a network with only IPv6 nodes and therefore might not have an available IPv6 router ID.

For configuring EIGRP for IPv6, see the *Configuring EIGRP for IPv6* section.

For more information about EIGRP for IPv6, see the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

EIGRP Features

EIGRP offers these features:

- Fast convergence.
- Incremental updates when the state of a destination changes, instead of sending the entire contents of the routing table, minimizing the bandwidth required for EIGRP packets.
- Less CPU usage because full update packets need not be processed each time that they are received.
- Protocol-independent neighbor discovery mechanism to learn about neighboring routers.
- Variable-length subnet masks (VLSMs).
- Arbitrary route summarization.
- EIGRP scales to large networks.

EIGRP Components

EIGRP has these four basic components:

- Neighbor discovery and recovery is the process that routers use to dynamically learn of other routers on their directly attached networks. Routers must also discover when their neighbors become unreachable or inoperative. Neighbor discovery and recovery is achieved with low overhead by periodically sending small hello packets. As long as hello packets are received, the Cisco IOS software can learn that a neighbor is alive and functioning. When this status is determined, the neighboring routers can exchange routing information.
- The reliable transport protocol is responsible for guaranteed, ordered delivery of EIGRP packets to all neighbors. It supports intermixed transmission of multicast and unicast packets. Some EIGRP packets must be sent reliably, and others need not be. For efficiency, reliability is provided only when necessary. For example, on a multiaccess network that has multicast capabilities (such as Ethernet), it is not necessary to send hellos reliably to all neighbors individually. Therefore, EIGRP sends a single multicast hello with an indication in the packet informing the receivers that the packet need not be acknowledged. Other types of packets (such as updates) require acknowledgment, which is shown in the packet. The reliable transport has a provision to send multicast packets quickly when there are unacknowledged packets pending. Doing so helps ensure that convergence time remains low in the presence of varying speed links.
- The DUAL finite state machine embodies the decision process for all route computations. It tracks all routes that are advertised by all neighbors. DUAL uses the distance information (known as a metric) to select efficient, loop-free paths. DUAL selects routes to be inserted into a routing table based on feasible successors. A successor is a neighboring router that is used for packet forwarding that has a least-cost path to a destination that is guaranteed not to be part of a routing loop. When there are no feasible successors, but there are neighbors advertising the destination, a recomputation must occur. This is the process whereby a new successor is determined. The amount of time it takes to recompute the route affects the convergence time. Recomputation is processor-intensive; it is advantageous to avoid recomputation if it is not necessary. When a topology change occurs, DUAL tests for feasible successors. If there are feasible successors, it uses any it finds to avoid unnecessary recomputation.
- The protocol-dependent modules are responsible for network layer protocol-specific tasks. An example is the IP EIGRP module, which is responsible for sending and receiving EIGRP packets that are

encapsulated in IP. It is also responsible for parsing EIGRP packets and informing DUAL of the new information received. EIGRP asks DUAL to make routing decisions, but the results are stored in the IP routing table. EIGRP is also responsible for redistributing routes that are learned by other IP routing protocols.

EIGRP Stub Routing

The EIGRP stub routing feature improves network stability, reduces resource utilization, and simplifies the stub device configuration.

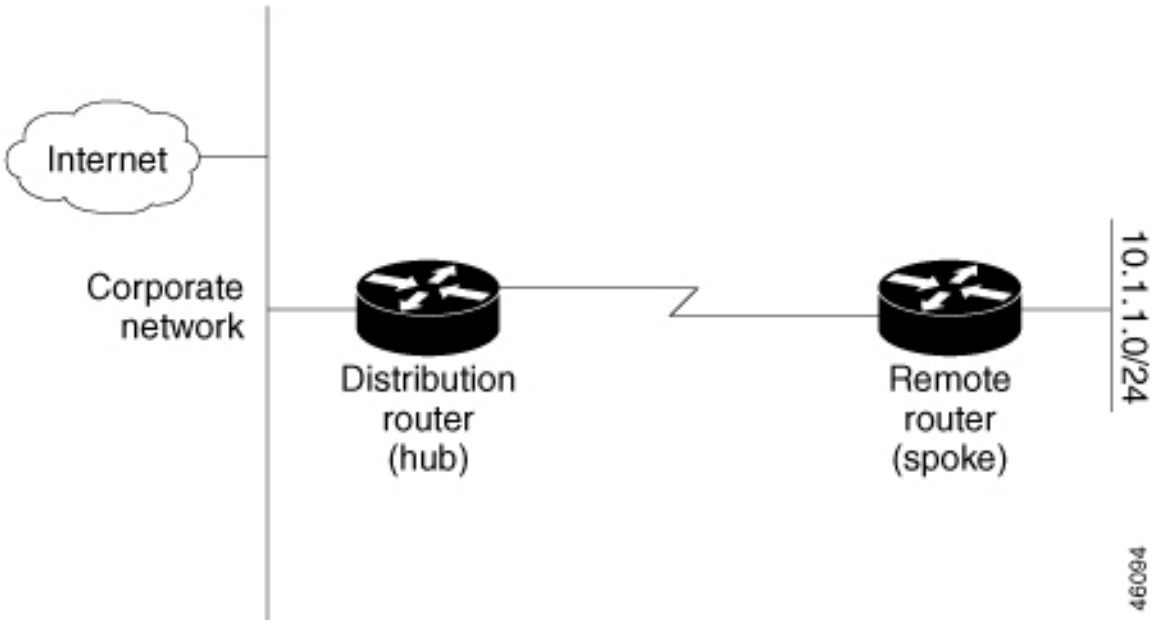
Stub routing is commonly used in hub-and-spoke network topologies. In a hub-and-spoke network, one or more end (stub) networks are connected to a remote device (the spoke) that is connected to one or more distribution devices (the hub). The remote device is adjacent to one or more distribution devices. The only route for IP traffic to reach the remote device is through a distribution device. This type of configuration is commonly used in WAN topologies, where the distribution device is directly connected to a WAN. The distribution device can be connected to many remote devices, which is often the case. In a hub-and-spoke topology, the remote device must forward all nonlocal traffic to a distribution device, so it becomes unnecessary for the remote device to have a complete routing table. Generally, the distribution device need not send anything more than a default route to the remote device.

When using the EIGRP stub routing feature, you need to configure the distribution and remote devices to use EIGRP and configure only the remote device as a stub. Only specified routes are propagated from the remote (stub) device. The stub device responds to all queries for summaries, connected routes, redistributed static routes, external routes, and internal routes with the message “inaccessible.” A device that is configured as a stub will send a special peer information packet to all neighboring devices to report its status as a stub device.

Any neighbor that receives a packet informing it of the stub status will not query the stub device for any routes, and a device that has a stub peer will not query that peer. The stub device will depend on the distribution device to send proper updates to all peers.

The figure below shows a simple hub-and-spoke network.

Figure 43: Simple Hub-and-Spoke Network



The stub routing feature by itself does not prevent routes from being advertised to the remote device. In the above example, the remote device can access the corporate network and the Internet only through the distribution device. Having a complete route table on the remote device would serve no functional purpose because the path to the corporate network and the Internet would always be through the distribution device. The large route table would only reduce the amount of memory that is required by the remote device. Bandwidth and memory can be conserved by summarizing and filtering routes in the distribution device. The remote device need not receive routes that have been learned from other networks because the remote device must send all nonlocal traffic, regardless of the destination, to the distribution device. If a true stub network is desired, the distribution device should be configured to send only a default route to the remote device. The EIGRP stub routing feature does not automatically enable summarization on distribution devices. In most cases, the network administrator will need to configure summarization on distribution devices.

Note When configuring the distribution device to send only a default route to the remote device, you must use the **ip classless** command on the remote device. By default, the **ip classless** command is enabled in all Cisco images that support the EIGRP stub routing feature.

Without the EIGRP stub routing feature, even after routes that are sent from the distribution device to the remote device have been filtered or summarized, a problem might occur. If a route is lost somewhere in the corporate network, EIGRP could send a query to the distribution device, which in turn would send a query to the remote device, even if routes are being summarized. If there is a communication problem (over the WAN link) between the distribution device and the remote device, an EIGRP stuck in active (SIA) condition could occur and cause instability elsewhere in the network. The EIGRP stub routing feature allows a network administrator to prevent queries from being sent to the remote device.

EIGRPv6 Stub Routing

The EIGRPv6 stub routing feature, reduces resource utilization by moving routed traffic closer to the end user.

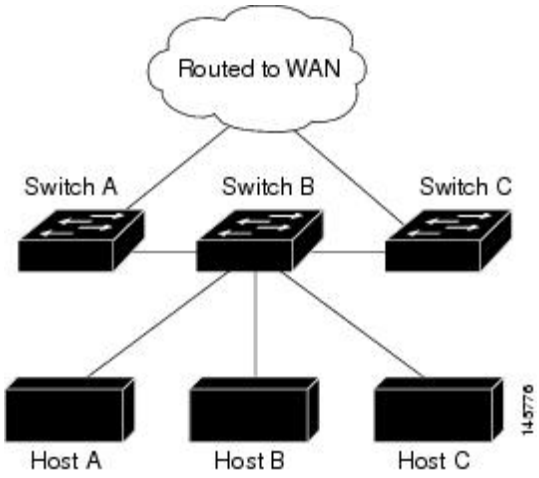
In a network using EIGRPv6 stub routing, the only allowable route for IPv6 traffic to the user is through a switch that is configured with EIGRPv6 stub routing. The switch sends the routed traffic to interfaces that are configured as user interfaces or are connected to other devices.

When using EIGRPv6 stub routing, you need to configure the distribution and remote routers to use EIGRPv6 and to configure only the switch as a stub. Only specified routes are propagated from the switch. The switch responds to all queries for summaries, connected routes, and routing updates.

Any neighbor that receives a packet informing it of the stub status does not query the stub router for any routes, and a router that has a stub peer does not query that peer. The stub router depends on the distribution router to send the proper updates to all peers.

In the figure given below, switch B is configured as an EIGRPv6 stub router. Switches A and C are connected to the rest of the WAN. Switch B advertises connected, static, redistribution, and summary routes to switch A and C. Switch B does not advertise any routes learned from switch A (and the reverse).

Figure 44: EIGRP Stub Router Configuration



For more information about EIGRPv6 stub routing, see “Implementing EIGRP for IPv6” section of the *Cisco IOS IP Configuration Guide, Volume 2 of 3: Routing Protocols, Release 12.4*.

How to Configure EIGRP

To create an EIGRP routing process, you must enable EIGRP and associate networks. EIGRP sends updates to the interfaces in the specified networks. If you do not specify an interface network, it is not advertised in any EIGRP update.

Note If you have devices on your network that are configured for IGRP, and you want to change to EIGRP, you must designate transition devices that have both IGRP and EIGRP configured. In these cases, perform Steps 1 through 3 in the next section and also see the "Configuring Split Horizon" section. You must use the same AS number for routes to be automatically redistributed.

Default EIGRP Configuration

Table 46: Default EIGRP Configuration

Feature	Default Setting
Auto summary	Disabled.
Default-information	Exterior routes are accepted and default information is passed between EIGRP processes when doing redistribution.
Default metric	Only connected routes and interface static routes can be redistributed without a default metric. The metric includes: • Bandwidth: 0 or greater kb/s. • Delay (tens of microseconds): 0 or any positive number that is a multiple of 39.1 nanoseconds. • Reliability: any number between 0 and 255 (255 means 100 percent reliability). • Loading: effective bandwidth as a number between 0 and 255 (255 is 100 percent loading). • MTU: maximum transmission unit size of the route in bytes. 0 or any positive integer.
Distance	Internal distance: 90. External distance: 170.
EIGRP log-neighbor changes	Disabled. No adjacency changes logged.
IP authentication key-chain	No authentication provided.
IP authentication mode	No authentication provided.
IP bandwidth-percent	50 percent.
IP hello interval	For low-speed nonbroadcast multiaccess (NBMA) networks: 60 seconds; all other networks: 5 seconds.
IP hold-time	For low-speed NBMA networks: 180 seconds; all other networks: 15 seconds.
IP split-horizon	Enabled.

Feature	Default Setting
IP summary address	No summary aggregate addresses are predefined.
Metric weights	tos: 0; k1 and k3: 1; k2, k4, and k5: 0
Network	None specified.
Offset-list	Disabled.
Router EIGRP	Disabled.
Set metric	No metric set in the route map.
Traffic-share	Distributed proportionately to the ratios of the metrics.
Variance	1 (equal-cost load-balancing).

Configuring Basic EIGRP Parameters

To configure basic EIGRP parameters, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**router eigrp** autonomous-system **Example:** `Device(config)#router eigrp 10`	Enables an EIGRP routing process, and enter router configuration mode. The AS number identifies the routes to other EIGRP devices and is used to tag routing information.
Step 4	**network** network-number **Example:** `Device(config-router)#network 192.168.0.0`	Associate networks with an EIGRP routing process. EIGRP sends updates to the interfaces in the specified networks.
Step 5	**eigrp log-neighbor-changes** **Example:**	(Optional) Enables logging of EIGRP neighbor changes to monitor routing system stability.

	Command or Action	Purpose
	`Device(config-router)#eigrp log-neighbor-changes`	
Step 6	**metric weights** *tos k1 k2 k3 k4 k5* **Example:** `Device(config-router)#metric weights 0 2 0 2 0 0`	(Optional) Adjust the EIGRP metric. Although the defaults have been carefully set to provide excellent operation in most networks, you can adjust them. **Caution** Setting metrics is complex and is not recommended without guidance from an experienced network designer.
Step 7	**offset-list** [*access-list number* | *name*] {**in** | **out**} *offset* [*type number*] **Example:** `Device(config-router)#offset-list 21 out 10`	(Optional) Applies an offset list to routing metrics to increase incoming and outgoing metrics to routes learned through EIGRP. You can limit the offset list with an access list or an interface.
Step 8	**auto-summary** **Example:** `Device(config-router)#auto-summary`	(Optional) Enables automatic summarization of subnet routes into network-level routes.
Step 9	**interface** *interface-id* **Example:** `Device(config-router)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 10	**ip summary-address eigrp** *autonomous-system-number address mask* **Example:** `Device(config-if)#ip summary-address eigrp 1 192.168.0.0 255.255.0.0`	(Optional) Configures a summary aggregate.
Step 11	**end** **Example:** `Device(config-if)#end`	Returns to privileged EXEC mode.
Step 12	**show ip protocols** **Example:** `Device#show ip protocols`	Verifies your entries.

	Command or Action	Purpose
Step 13	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring EIGRP Interfaces

Other optional EIGRP parameters can be configured on an interface basis.

To configure EIGRP interfaces, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 4	**ip bandwidth-percent eigrp** *percent* **Example:** `Device(config-if)#ip bandwidth-percent eigrp 60`	(Optional) Configures the percentage of bandwidth that can be used by EIGRP on an interface. The default is 50 percent.
Step 5	**ip summary-address eigrp** *autonomous-system-number address mask* **Example:** `Device(config-if)#ip summary-address eigrp 109 192.161.0.0 255.255.0.0`	(Optional) Configures a summary aggregate address for a specified interface (not usually necessary if auto-summary is enabled).

	Command or Action	Purpose
Step 6	**ip hello-interval eigrp** *autonomous-system-number seconds* **Example:** `Device(config-if)#ip hello-interval eigrp 109 10`	(Optional) Change the hello time interval for an EIGRP routing process. The range is 1 to 65535 seconds. The default is 60 seconds for low-speed NBMA networks and 5 seconds for all other networks.
Step 7	**ip hold-time eigrp** *autonomous-system-number seconds* **Example:** `Device(config-if)#ip hold-time eigrp 109 40`	(Optional) Change the hold time interval for an EIGRP routing process. The range is 1 to 65535 seconds. The default is 180 seconds for low-speed NBMA networks and 15 seconds for all other networks. **Caution** Do not adjust the hold time without consulting Cisco technical support.
Step 8	**no ip split-horizon eigrp** *autonomous-system-number* **Example:** `Device(config-if)#no ip split-horizon eigrp 109`	(Optional) Disables split horizon to allow route information to be advertised by a router out any interface from which that information originated.
Step 9	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 10	**show ip eigrp interface** **Example:** `Device#show ip eigrp interface`	Displays which interfaces EIGRP is active on and information about EIGRP relating to those interfaces.
Step 11	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring EIGRP for IPv6

Before configuring the switch to run IPv6 EIGRP, enable routing by entering the **ip routing global configuration** command, enable the forwarding of IPv6 packets by entering the **ipv6 unicast-routing global** configuration command, and enable IPv6 on any Layer 3 interfaces on which you want to enable IPv6 EIGRP.

To set an explicit router ID, use the **show ipv6 eigrp** command to see the configured router IDs, and then use the **router-id** command.

As with EIGRP IPv4, you can use EIGRPv6 to specify your EIGRP IPv6 interfaces, and to select a subset of those as passive interfaces. Use the **passive-interface** command to make an interface passive, and then use the **no passive-interface** command on selected interfaces to make them active. EIGRP IPv6 does not need to be configured on a passive interface.

For more configuration procedures, see the "Implementing EIGRP for IPv6" chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

Configuring EIGRP Route Authentication

EIGRP route authentication provides MD5 authentication of routing updates from the EIGRP routing protocol to prevent the introduction of unauthorized or false routing messages from unapproved sources.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** interface-id **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 4	**ip authentication mode eigrp** *autonomous-system* **md5** **Example:** `Device(config-if)#ip authentication mode eigrp 104 md5`	Enables MD5 authentication in IP EIGRP packets.
Step 5	**ip authentication key-chain eigrp** *autonomous-system key-chain* **Example:** `Device(config-if)#ip authentication key-chain eigrp 105 chain1`	Enables authentication of IP EIGRP packets.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Device(config-if)#exit`	Returns to global configuration mode.
Step 7	**key chain** *name-of-chain* **Example:** `Device(config)#key chain chain1`	Identify a key chain and enter key-chain configuration mode. Match the name configured in Step 4.
Step 8	**key** *number* **Example:** `Device(config-keychain)#key 1`	In key-chain configuration mode, identify the key number.
Step 9	**key-string** *text* **Example:** `Device(config-keychain-key)#key-string key1`	In key-chain key configuration mode, identify the key string.
Step 10	**accept-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*} **Example:** `Device(config-keychain-key)#accept-lifetime 13:30:00 Jan 25 2011 duration 7200`	(Optional) Specifies the time period during which the key can be received. The *start-time* and *end-time* syntax can be either *hh:mm:ss Month date year* or *hh:mm:ss date Month year*. The default is forever with the default *start-time* and the earliest acceptable date as January 1, 1993. The default *end-time* and **duration** is **infinite**.
Step 11	**send-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*} **Example:** `Device(config-keychain-key)#send-lifetime 14:00:00 Jan 25 2011 duration 3600`	(Optional) Specifies the time period during which the key can be sent. The *start-time* and *end-time* syntax can be either *hh:mm:ss Month date year* or *hh:mm:ss date Month year*. The default is forever with the default *start-time* and the earliest acceptable date as January 1, 1993. The default *end-time* and **duration** is **infinite**.
Step 12	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 13	**show key chain** **Example:**	Displays authentication key information.

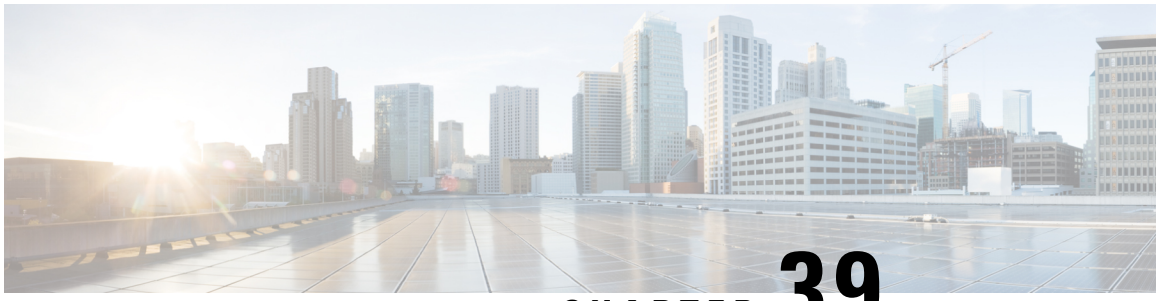
	Command or Action	Purpose
	`Device#show key chain`	
Step 14	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring and Maintaining EIGRP

You can delete neighbors from the neighbor table. You can also display various EIGRP routing statistics. The table given below lists the privileged EXEC commands for deleting neighbors and displaying statistics.

Table 47: IP EIGRP Clear and Show Commands

Command	Purpose
clear ip eigrp neighbors [*if-address* | *interface*]	Deletes neighbors from the neighbor table.
show ip eigrp interface [*interface*] [*as number*]	Displays information about interfaces configured for EIGRP.
show ip eigrp neighbors [*type-number*]	Displays EIGRP discovered neighbors.
show ip eigrp topology [*autonomous-system-number*] | [[*ip-address*] *mask*]]	Displays the EIGRP topology table for a given process.
show ip eigrp traffic [*autonomous-system-number*]	Displays the number of packets sent and received for all or a specified EIGRP process.



CHAPTER 39

Configuring EIGRP Prefix Limit Support

Prerequisites for EIGRP Prefix Limit Support

- Multiprotocol Label Switching (MPLS) Virtual Private Network (VPN) services have been configured between the Provider Edge (PE) routers and the customer edge (CE) routers at the customer sites.

Restrictions for EIGRP Prefix Limit Support

- This feature is supported only under the IPv4 VRF address family and can be used only to limit the number of prefixes that are accepted through a VRF.
- The EIGRP Prefix Limiting Support feature is enabled only under the IPv4 VRF address-family. A peer that is configured to send too many prefixes or a peer that rapidly advertises and then withdraws prefixes can cause instability in the network. This feature can be configured to automatically reestablish a disabled peering session at the default or user-defined time interval or when the maximum-prefix limit is not exceeded. However, the configuration of this feature alone cannot change or correct a peer that is sending an excessive number of prefixes. If the maximum-prefix limit is exceeded, you will need to reconfigure the maximum-prefix limit or reduce the number of prefixes that are sent from the peer.

Information About EIGRP Prefix Limit Support

The EIGRP Prefix Limit Support feature introduces the capability to limit the number of prefixes per VRF that are accepted from a specific peer or to limit all prefixes that are accepted by an Enhanced Interior Gateway Routing Protocol (EIGRP) process through peering and redistribution. This feature is designed to protect the local device from external misconfiguration that can negatively impact local system resources; for example, a peer that is misconfigured to redistribute full Border Gateway Protocol (BGP) routing tables into EIGRP.

This feature is enabled under the IPv4 VRF address family and can be configured to support the MPLS VPN Support for EIGRP Between Provider Edge and Customer Edge feature.

The EIGRP Prefix Limit Support feature provides the ability to configure a limit on the number of prefixes that are accepted from EIGRP peers or learned through redistribution. This feature can be configured on per-peer or per-process basis and can be configured for all peers and processes. This feature is designed to protect the local device from misconfigured external peers by limiting the amount of system resources that can be consumed to process prefix updates.

Misconfigured VPN Peers

In MPLS VPNs, the number of routes that are permitted in the VPN routing and forwarding instance (VRF) is configured with the **maximum routes** VRF configuration command. However, limiting the number routes permitted in the VPN does not protect the local device from a misconfigured peer that sends an excessive number of routes or prefixes. This type of external misconfiguration can have a negative effect on the local device by consuming all available system resources (CPU and memory) in processing prefix updates. This type of misconfiguration can occur on a peer that is not within the control of the local administrator.

Protecting the Device from External Peers

This feature can be configured to protect an individual peering session or protect all peering sessions. When this feature is enabled and the maximum-prefix limit has been exceeded, the device will tear down the peering session, clear all routes that were learned from the peer, and then place the peer in a penalty state for the default or user-defined time period. After the penalty time period expires, normal peering will be reestablished.

Limiting the Number of Redistributed Prefixes

This feature can be configured to limit the number of prefixes that are accepted into the EIGRP topology table through redistribution from the Routing Information Base (RIB). All sources of redistribution are processed cumulatively. When the maximum-prefix limit is exceeded, all routes learned through redistribution are discarded and redistribution is suspended for the default or user-defined time period. After the penalty time period expires, normal redistribution will occur.

Protecting the Device at the EIGRP Process Level

This feature can be configured to protect the device at the EIGRP process level. When this feature is configured at the EIGRP process level, the maximum-prefix limit is applied to all peering sessions and to route redistribution. When the maximum-prefix limit is exceeded, all sessions with the remote peers are torn down, all routes learned from remote peers are removed from the topology and routing tables, all routes learned through redistribution are discarded, and redistribution and peering are suspended for the default or user-defined time period.

Warning-Only Mode

The EIGRP Prefix Limit Support feature has two modes of operation. This feature can control peering and redistribution per default and user-defined values or this feature can operate in warning-only mode. In warning-only mode the device will monitor the number of prefixes learned through peering and/or redistribution but will not take any action when the maximum-prefix limit is exceeded. Warning-only mode is activated only when the **warning-only** keyword is configured for any of the maximum-prefix limit commands. Only

syslog messages are generated when this mode of operation is enabled. Syslog messages can be sent to a syslog server or printed in the console. These messages can be buffered or rate limited per standard Cisco IOS system logging configuration options.

Restart Reset and Dampening Timers and Counters

The EIGRP Prefix Limit Support feature provides two user-configurable timers, a restart counter, and a dampening mechanism. When the maximum-prefix limit is exceeded, peering and/or redistribution is suspended for a default or user-defined time period. If the maximum-prefix limit is exceeded too often, redistribution and/or peering will be suspended until manual intervention is taken.

Restart Timer

The restart timer determines how long the router will wait to form an adjacency or accept redistributed routes from the RIB after the maximum-prefix limit has been exceeded. The default restart-time period is 5 minutes.

Restart Counter

The restart counter determines the number of times a peering session can be automatically reestablished after the peering session has been torn down or after the a redistributed routes have been cleared and relearned because the maximum-prefix limit has been exceeded. The default restart-count limit is three.

Note After the restart count limit has been crossed, you will need to enter the **neighbor**, or **clear eigrp address-family neighbor** command to restore normal peering and redistribution.

Reset Timer

The reset timer is used to configure the device to reset the restart count to 0 after the default or configured reset-time period has expired. This timer is designed to provide administrator with control over long-and medium-term accumulated penalties. The default reset-time period is 15 minutes.

Dampening Mechanism

The dampening mechanism is used to apply an exponential decay penalty to the restart-time period each time the maximum-prefix limit is exceeded. The half-life for the decay penalty is 150 percent of the default or user-defined restart-time value in minutes. This mechanism is designed to identify and suppress unstable peers. It is disabled by default.

How to Configure the Maximum-Prefix Limit

Note If the EIGRP process enters into a suspended (pending or down) state, the device will not establish neighborships with new peers and thus cease to transmit and stop processing hello packets.

Configuring the Maximum Number of Prefix Accepted from Peering Sessions Autonomous System Configuration

The maximum-prefix limit can be configured for all peering sessions or individual peering sessions with the **neighbor maximum-prefix**(EIGRP) command. When the maximum-prefix limit is exceeded, the session with the remote peer is torn down and all routes learned from the remote peer are removed from the topology and routing tables. The maximum-prefix limit that can be configured is limited only by the available system resources on the device.

Note In EIGRP, **neighbor** commands have been used traditionally to configure static neighbors. In the context of this feature, however, the **neighbor maximum-prefix** command can be used to configure the maximum-prefix limit for both statically configured and dynamically discovered neighbors.

Default or user-defined restart, restart-count, and reset-time values for the process-level configuration of this feature, configured with the **maximum-prefix** command, are inherited by the **redistribute maximum-prefix** and **neighbor maximum-prefix** command configurations by default. If a single peer is configured with the **neighbor maximum-prefix** command, a process-level configuration or a configuration that is applied to all neighbors will be inherited.

Note

- VRFs have been created and configured.
- EIGRP peering is established through the MPLS VPN.
- This task can be configured only in IPv4 VRF address family configuration mode.
- When you configure the **neighbor maximum-prefix** command to protect a single peering session, only the maximum-prefix limit, the percentage threshold, the warning-only configuration options can be configured. Session dampening, restart, and reset timers are configured on a global basis.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router eigrp** *as-number* **Example:**	Enters router configuration mode and creates an EIGRP routing process.

	Command or Action	Purpose
	`Device(config)# router eigrp 1`	• A maximum of 30 EIGRP routing processes can be configured.
Step 4	**address-family ipv4** [**multicast**][**unicast**][**vrf** *vrf-name*] **autonomous-system** *autonomous-system-number* **Example:** `Device(config-router)# address-family ipv4 vrf vrf1`	Enters address family configuration mode and creates a session for the VRF.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **description** *text* **Example:** `Device(config-router-af)# neighbor 172.16.2.3 description peer with example.com`	(Optional) Associates a description with a neighbor.
Step 6	**neighbor** *ip-address* **maximum-prefix** *maximum* [*threshold*] [**warning-only**] **Example:** `Device(config-router-af)# neighbor 10.0.0.1 maximum-prefix 10000 80 warning-only`	Limits the number of prefixes that are accepted from the specified EIGRP neighbor.
Step 7	**neighbor maximum-prefix** *maximum* [*threshold*] [[**dampened**] [**reset-time** *minutes*] [**restart** *minutes*] [**restart-count** *number*] | **warning-only**] **Example:** `Device(config-router-af)# neighbor maximum-prefix 10000 80 warning-only`	Limits the number of prefixes that are accepted from all EIGRP neighbors.
Step 8	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.

Configuring the Maximum Number of Prefixes Accepted from Peering Sessions Named Configuration

The maximum-prefix limit can be configured for all peering sessions or individual peering sessions with the **neighbor maximum-prefix**(EIGRP) command. When the maximum-prefix limit is exceeded, the session with the remote peer is torn down and all routes learned from the remote peer are removed from the topology

and routing tables. The maximum-prefix limit that can be configured is limited only by the available system resources on the device.

Note In EIGRP, **neighbor** commands have been used traditionally to configure static neighbors. In the context of this feature, however, the **neighbor maximum-prefix** command can be used to configure the maximum-prefix limit for both statically configured and dynamically discovered neighbors.

Default or user-defined restart, restart-count, and reset-time values for the process-level configuration of this feature, configured with the **maximum-prefix** command, are inherited by the **redistribute maximum-prefix** and **neighbor maximum-prefix** command configurations by default. If a single peer is configured with the **neighbor maximum-prefix** command, a process-level configuration or a configuration that is applied to all neighbors will be inherited.

Note

- VRFs have been created and configured.
- EIGRP peering is established through the MPLS VPN.
- This task can be configured only in IPv4 VRF address family configuration mode.
- When you configure the **neighbor maximum-prefix** command to protect a single peering session, only the maximum-prefix limit, the percentage threshold, the warning-only configuration options can be configured. Session dampening, restart, and reset timers are configured on a global basis.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router eigrp** *virtual-instance-name* **Example:** `Device(config)# router eigrp virtual-name1`	Enters router configuration mode and creates an EIGRP routing process. • A maximum of 30 EIGRP routing processes can be configured.
Step 4	**address-family ipv4** [**multicast**] [**unicast**] [**vrf** *vrf-name*] **autonomous-system** *autonomous-system-number*	Enters address family configuration mode and creates a session for the VRF.

	Command or Action	Purpose
	Example: `Device(config-router)# address-family ipv4 vrf RED autonomous-system 45000`	
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **description** *text* **Example:** `Device(config-router-af)# neighbor 172.16.2.3 description peer with example.com`	(Optional) Associates a description with a neighbor.
Step 6	**neighbor** *ip-address* **maximum-prefix** *maximum* [*threshold*][**warning-only**] **Example:** `Device(config-router-af)# neighbor 10.0.0.1 maximum-prefix 10000 80 warning-only`	Limits the number of prefixes that are accepted from the specified EIGRP neighbor.
Step 7	**neighbor maximum-prefix** *maximum* [*threshold*] [[**dampened**] [**reset-time** *minutes*] [**restart** *minutes*] [**restart-count** *number*] | **warning-only**] **Example:** `Device(config-router-af)# neighbor maximum-prefix 10000 80 warning-only`	Limits the number of prefixes that are accepted from all EIGRP neighbors.
Step 8	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode.

Configuring the Maximum Number of Prefixes Learned Through Redistribution Autonomous System Configuration

The maximum-prefix limit can be configured for prefixes learned through redistribution with the **redistribute maximum-prefix** (EIGRP) command. When the maximum-prefix limit is exceeded, all routes learned from the RIB will be discarded and redistribution will be suspended for the default or user-defined time period. The maximum-prefix limit that can be configured for redistributed prefixes is limited only by the available system resources on the device.

Default or user-defined restart, restart-count, and reset-time values for the process-level configuration of this feature, configured with the **maximum-prefix** command, are inherited by the **redistribute maximum-prefix** and **neighbor maximum-prefix** command configurations by default. If a single peer is configured with the

neighbor maximum-prefix command, a process-level configuration or a configuration that is applied to all neighbors will be inherited.

Note

- VRFs have been created and configured.
- EIGRP peering is established through the MPLS VPN.
- This task can be configured only in IPv4 VRF address family configuration mode.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router eigrp** *as-number* **Example:** `Device(config)# router eigrp 1`	Enters router configuration mode and creates an EIGRP routing process. • A maximum of 30 EIGRP routing processes can be configured.
Step 4	**address-family ipv4** [**unicast**] **vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4 vrf VRF1`	Enters address family configuration mode and creates a session for the VRF.
Step 5	**redistribute maximum-prefix** *maximum* [*threshold*] [[**dampened**] [**reset-time** *minutes*] [**restart** *minutes*] [**restart-count** *number*] | **warning-only**] **Example:** `Device(config-router-af)# redistribute maximum-prefix 10000 80 reset-time 10 restart 2`	Limits the number of prefixes redistributed into an EIGRP process.
Step 6	**end** **Example:**	Exits address family configuration mode and enters privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-router-af)# end`	

Configuring the Maximum Number of Prefixes Learned Through Redistribution Named Configuration

The maximum-prefix limit can be configured for prefixes learned through redistribution with the **redistribute maximum-prefix**(EIGRP) command. When the maximum-prefix limit is exceeded, all routes learned from the RIB will be discarded and redistribution will be suspended for the default or user-defined time period. The maximum-prefix limit that can be configured for redistributed prefixes is limited only by the available system resources on the device.

Default or user-defined restart, restart-count, and reset-time values for the process-level configuration of this feature, configured with the **maximum-prefix** command, are inherited by the **redistribute maximum-prefix** and **neighbor maximum-prefix** command configurations by default. If a single peer is configured with the **neighbor maximum-prefix** command, a process-level configuration or a configuration that is applied to all neighbors will be inherited.

Note

- VRFs have been created and configured.
- EIGRP peering is established through the MPLS VPN.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router eigrp** *virtual-instance-name* **Example:** `Device(config)# router eigrp virtual-name1`	Enters router configuration mode and creates an EIGRP routing process. • A maximum of 30 EIGRP routing processes can be configured.
Step 4	**address-family ipv4** [**multicast**] [**unicast**] [**vrf** *vrf-name*] **autonomous-system** *autonomous-system-number*	Enters address family configuration mode and creates a session for the VRF.

	Command or Action	Purpose
	Example: `Device(config-router)# address-family ipv4 vrf VRF1`	
Step 5	**network** *ip-address* [*wildcard-mask*] **Example:** `Device(config-router-af)# network 172.16.0.0`	Specifies the network for an EIGRP address family routing process.
Step 6	**topology** {**base** | *topology-name* **tid** *number*} **Example:** `Device(config-router-af)# topology base`	Configures an EIGRP process to route traffic under the specified topology instance and enters address-family topology configuration mode.
Step 7	**redistribute maximum-prefix** *maximum* [*threshold*] [[**dampened**] [**reset-time** *minutes*] [**restart** *minutes*] [**restart-count** *number*] | **warning-only**] **Example:** `Device(config-router-af-topology)# redistribute maximum-prefix 10000 80 reset-time 10 restart 2`	Limits the number of prefixes redistributed into an EIGRP process.
Step 8	**exit-af-topology** **Example:** `Device(config-router-af-topology)# exit-af-topology`	Exits address family topology configuration mode.

Configuring the Maximum-Prefix Limit for an EIGRP Process Autonomous System Configuration

The maximum-prefix limit can be configured for an EIGRP process to limit the number prefixes that are accepted from all sources. This task is configured with the **maximum-prefix**command. When the maximum-prefix limit is exceeded, sessions with the remote peers are brought down and all routes learned from remote peers are removed from the topology and routing tables. Also, all routes learned from the RIB are discarded and redistribution is suspended for the default or user-defined time period.

Default or user-defined restart, restart-count, and reset-time values for the process-level configuration of this feature, configured with the **maximum-prefix** command, are inherited by the **redistribute maximum-prefix** and **neighbor maximum-prefix** command configurations by default. If a single peer is configured with the **neighbor maximum-prefix** command, a process-level configuration or a configuration that is applied to all neighbors will be inherited.

Note

- VRFs have been created and configured.
- EIGRP peering is established through the MPLS VPN.
- This task can be configured only in IPv4 VRF address family configuration mode.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router eigrp** *as-number* **Example:** `Device(config)# router eigrp 1`	Enters router configuration mode and creates an EIGRP routing process. • A maximum of 30 EIGRP routing processes can be configured.
Step 4	**address-family ipv4** [**unicast**] **vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4 vrf RED`	Enters address family configuration mode and creates a session for the VRF.
Step 5	**maximum-prefix** *maximum* [*threshold*] [[**dampened**] [**reset-time** *minutes*] [**restart** *minutes*] [**restart-count** *number*] | **warning-only**] **Example:** `Device(config-router-af)# maximum-prefix 10000 80 reset-time 10 restart 2 warning-only`	Limits the number of prefixes that are accepted under an address family by an EIGRP process. • The example configures a maximum-prefix limit of 10,000 prefixes, a reset time period of 10 minutes, a warning message to be displayed at 80 percent of the maximum-prefix limit, and a restart time period of 2 minutes.
Step 6	**end** **Example:** `Device(config-router-af)# end`	Exits address-family configuration mode and enters privileged EXEC mode.

Configuring the Maximum-Prefix Limit for an EIGRP Process Named Configuration

The maximum-prefix limit can be configured for an EIGRP process to limit the number prefixes that are accepted from all sources. This task is configured with the **maximum-prefix**command. When the maximum-prefix limit is exceeded, sessions with the remote peers are brought down and all routes learned from remote peers are removed from the topology and routing tables. Also, all routes learned from the RIB are discarded and redistribution is suspended for the default or user-defined time period.

Default or user-defined restart, restart-count, and reset-time values for the process-level configuration of this feature, configured with the **maximum-prefix** command, are inherited by the **redistribute maximum-prefix** and **neighbor maximum-prefix** command configurations by default. If a single peer is configured with the **neighbor maximum-prefix** command, a process-level configuration or a configuration that is applied to all neighbors will be inherited.

Note

- VRFs have been created and configured.
- EIGRP peering is established through the MPLS VPN.
- This task can be configured only in IPv4 VRF address family configuration mode.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router eigrp** *virtual-instance-name* **Example:** `Device(config)# router eigrp virtual-name1`	Creates an EIGRP routing process and enters router configuration mode. • A maximum of 30 EIGRP routing processes can be configured.
Step 4	**address-family ipv4** [**multicast**] [**unicast**] [**vrf** *vrf-name*] **autonomous-system** *autonomous-system-number* **Example:** `Device(config-router)# address-family ipv4 vrf VRF1 autonomous-system 45000`	Enters address family configuration mode and creates a session for the VRF.

	Command or Action	Purpose
Step 5	**topology** {**base** | *topology-name* **tid** *number*} **Example:** `Device(config-router-af)# topology base`	Configures an EIGRP process to route traffic under the specified topology instance and enters address family topology configuration mode.
Step 6	**maximum-prefix** *maximum* [*threshold*] [[**dampened**] [**reset-time** *minutes*] [**restart** *minutes*] [**restart-count** *number*] | **warning-only**] **Example:** `Device(config-router-af-topology)# maximum- prefix 10000 80 reset-time 10 restart 2 warning-only`	Limits the number of prefixes that are accepted under an address family by an EIGRP process. • The example configures a maximum-prefix limit of 10,000 prefixes, a reset time period of 10 minutes, a warning message to be displayed at 80 percent of the maximum-prefix limit, and a restart time period of 2 minutes.
Step 7	**exit-af-topology** **Example:** `Device(config-router-af-topology)# exit-af-topology`	Exits address family topology configuration mode.
Step 8	**show eigrp address-family** {**ipv4** | **ipv6**} [**vrf** *vrf-name*] [*autonomous-system-number*] [**multicast**] **accounting** **Example:** `Device# show eigrp address-family ipv4 22 accounting`	(Optional) Displays prefix accounting information for EIGRP processes. **Note** Connected and summary routes are not listed individually in the output from this **show** command but are counted in the total aggregate count per process.

Configuration Examples for Configuring the Maximum-Prefix Limit

Example Configuring the Maximum-Prefix Limit for a Single Peer--Autonomous System Configuration

The following example, starting in global configuration mode, configures the maximum-prefix limit for a single peer. The maximum limit is set to 1000 prefixes, and the warning threshold is set to 80 percent. When the maximum-prefix limit is exceeded, the session with this peer will be torn down, all routes learned from this peer will be removed from the topology and routing tables, and this peer will be placed in a penalty state for 5 minutes (default penalty value).

```
Device(config)# router eigrp 100
Device(config-router)# address-family ipv4 vrf VRF1
```

```
Device(config-router-af)# neighbor 10.0.0.1 maximum-prefix 1000 80
Device(config-router-af)# end
```

Note If the maximum prefix limit at process level and neighbor level is set together then the max prefix limit at process level will take precedence. When the max prefix limit at neighbor level is set greater than the max prefix limit set at process level, the device displays this message:

```
Max prefix limit at neighbor level is set to a value (%d) greater than max prefix limit at
 process level (%d)
```

Example Configuring the Maximum-Prefix Limit for a Single Peer--Named Configuration

The following example, starting in global configuration mode, configures the maximum-prefix limit for a single peer. The maximum limit is set to 1000 prefixes, and the warning threshold is set to 80 percent. When the maximum-prefix limit is exceeded, the session with this peer will be torn down, all routes learned from this peer will be removed from the topology and routing tables, and this peer will be placed in a penalty state for 5 minutes (default penalty value).

```
Device(config)# router eigrp virtual-name1
Device(config-router)# address-family ipv4 vrf VRF1 autonomous-system 45000
Device(config-router-af)# neighbor 10.0.0.1 maximum-prefix 1000 80

Device(config-router-af)# exit-address-family
```

Note If the maximum prefix limit at process level and neighbor level is set together then the max prefix limit at process level will take precedence.When the max prefix limit at neighbor level is set greater than the max prefix limit set at process level, the device displays this message:

```
Max prefix limit at neighbor level is set to a value (%d) greater than max prefix limit at
 process level (%d)
```

Example Configuring the Maximum-Prefix Limit for All Peers--Autonomous System Configuration

The following example, starting in global configuration mode, configures the maximum-prefix limit for all peers. The maximum limit is set to 10,000 prefixes, the warning threshold is set to 90 percent, the restart timer is set to 4 minutes, a decay penalty is configured for the restart timer with the **dampened**keyword, and all timers are configured to be reset to 0 every 60 minutes. When the maximum-prefix limit is exceeded, all peering sessions will be torn down, all routes learned from all peers will be removed from the topology and routing tables, and all peers will be placed in a penalty state for 4 minutes (user-defined penalty value). A dampening exponential decay penalty will also be applied.

```
Router(config)# router eigrp 100
Router(config-router)# address-family ipv4 vrf VRF1
Router(config-router-af)# neighbor maximum-prefix 10000 90 dampened reset-time 60 restart
```

```
4
Router(config-router-af)# end
```

Note If the maximum prefix limit at process level and neighbor level is set together then the max prefix limit at process level will take precedence. When the max prefix limit at neighbor level is set greater than the max prefix limit set at process level, the device displays this message:

```
Max prefix limit at neighbor level is set to a value (%d) greater than max prefix limit at
 process level (%d)
```

Example Configuring the Maximum-Prefix Limit for All Peers--Named Configuration

The following example, starting in global configuration mode, configures the maximum-prefix limit for all peers. The maximum limit is set to 10,000 prefixes, the warning threshold is set to 90 percent, the restart timer is set to 4 minutes, a decay penalty is configured for the restart timer with the **dampened**keyword, and all timers are configured to be reset to 0 every 60 minutes. When the maximum-prefix limit is exceeded, all peering sessions will be torn down, all routes learned from all peers will be removed from the topology and routing tables, and all peers will be placed in a penalty state for 4 minutes (user-defined penalty value). A dampening exponential decay penalty will also be applied.

```
Device(config)# router eigrp virtual-name1
Device(config-router)# address-family ipv4 vrf VRF1 autonomous-system 45000
Device(config-router-af)# neighbor maximum-prefix 10000 90 dampened reset-time 60 restart
4
Device(config-router-af)# exit-address-family
```

Note If the maximum prefix limit at process level and neighbor level is set together then the max prefix limit at process level will take precedence. When the max prefix limit at neighbor level is set greater than the max prefix limit set at process level, the device displays this message:

```
Max prefix limit at neighbor level is set to a value (%d) greater than max prefix limit at
 process level (%d)
```

Example Configuring the Maximum-Prefix Limit for Redistributed Routes--Autonomous System Configuration

The following example, starting in global configuration mode, configures the maximum-prefix limit for routes learned through redistribution. The maximum limit is set to 5000 prefixes and the warning threshold is set to 95 percent. When the number of prefixes learned through redistribution reaches 4750 (95 percent of 5000), warning messages will be displayed in the console. Because the **warning-only** keyword is configured, the topology and routing tables will not be cleared and route redistribution will not be placed in a penalty state.

```
Device(config)# router eigrp 100
Device(config-router)# address-family ipv4 vrf VRF1
Device(config-router-af)# redistribute maximum-prefix 5000 95 warning-only
Device(config-router-af)# end
```

Note When the maximum prefix limit is configured at both the process level and redistribution level, the limit set at the process level will take precedence. In cases where the max prefix limit at redistribution level is set greater than the max prefix limit set at process level, the device displays this message:

```
Max prefix limit at redistribute level is set to a value (%d) greater than max prefix limit
 at process level (%d)
```

Example Configuring the Maximum-Prefix Limit for Redistributed Routes--Named Configuration

The following example, starting in global configuration mode, configures the maximum-prefix limit for routes learned through redistribution. The maximum limit is set to 5000 prefixes and the warning threshold is set to 95 percent. When the number of prefixes learned through redistribution reaches 4750 (95 percent of 5000), warning messages will be displayed in the console. Because the **warning-only** keyword is configured, the topology and routing tables will not be cleared and route redistribution will not be placed in a penalty state.

```
Device(config)# router eigrp virtual-name1
Device(config-router)# address-family ipv4 vrf VRF1 autonomous-system 45000
Device(config-router-af)# network 172.16.0.0
Device(config-router-af)# topology base
Device(config-router-af-topology)# redistribute maximum-prefix 5000 95 warning-only
Device(config-router-af-topology)# exit-af-topology
```

Note When the maximum prefix limit is configured at both the process level and redistribution level, the limit set at the process level will take precedence. In cases where the max prefix limit at redistribution level is set greater than the max prefix limit set at process level, the device displays this message:

```
Max prefix limit at redistribute level is set to a value (%d) greater than max prefix limit
 at process level (%d)
```

Example Configuring the Maximum-Prefix Limit for an EIGRP Process--Autonomous System Configuration

The following example, starting in global configuration mode, configures the maximum-prefix limit for an EIGRP process, which includes routes learned through redistribution and routes learned through EIGRP peering sessions. The maximum limit is set to 50,000 prefixes. When the number of prefixes learned through redistribution reaches 37,500 (75 percent of 50,000), warning messages will be displayed in the console.

When the maximum-prefix limit is exceeded, all peering sessions will be reset, the topology and routing tables will be cleared, and redistributed routes and all peering sessions will be placed in a penalty state.

```
Device(config)# router eigrp 100
Device(config-router)# address-family ipv4 vrf RED
Device(config-router-af)# maximum-prefix 50000
Device(config-router-af)# end
```

Note

- When the max prefix limit at process level is set lower than the max prefix limit set at redistribute level, the device displays this message:

  ```
  Max prefix limit at process level is set to a value (%d) lower than max prefix limit
  at redistribute level (%d)
  ```

- When the max prefix limit at process level is set lower than the max prefix limit set at neighbor level, the device displays this message:

  ```
  (%d) lower than max prefix limit at neighbor level (%d)
  ```

Example Configuring the Maximum-Prefix Limit for an EIGRP Process--Named Configuration

The following example, starting in global configuration mode, configures the maximum-prefix limit for an EIGRP process, which includes routes learned through redistribution and routes learned through EIGRP peering sessions. The maximum limit is set to 50,000 prefixes. When the number of prefixes learned through redistribution reaches 37,500 (75 percent of 50,000), warning messages will be displayed in the console.

When the maximum-prefix limit is exceeded, all peering sessions will be reset, the topology and routing tables will be cleared, and redistributed routes and all peering sessions will be placed in a penalty state.

```
Device(config)# router eigrp virtual-name
Device(config-router)# address-family ipv4 vrf VRF1 autonomous-system 45000
Device(config-router-af)# topology base
Device(config-router-af-topology)# maximum-prefix 50000
Device(config-router-af-topology)# exit-af-topology
```

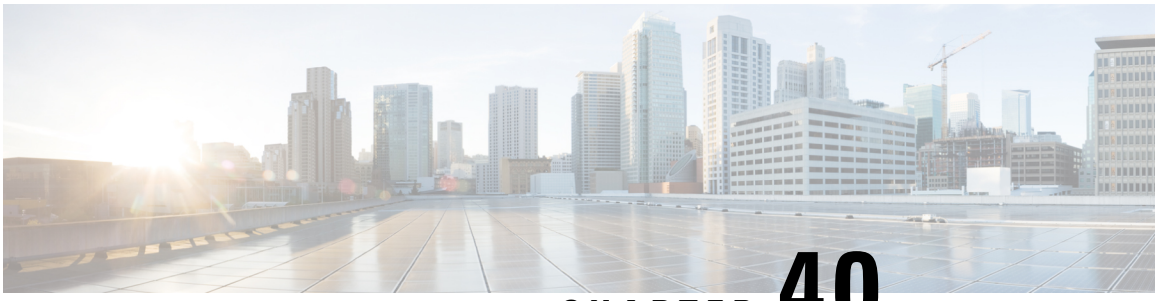
Note

- When the max prefix limit at process level is set lower than the max prefix limit set at redistribute level, the device displays this message:

  ```
  Max prefix limit at process level is set to a value (%d) lower than max prefix limit
  at redistribute level (%d)
  ```

- When the max prefix limit at process level is set lower than the max prefix limit set at neighbor level, the device displays this message:

  ```
  (%d) lower than max prefix limit at neighbor level (%d)
  ```



CHAPTER 40

Configuring BGP

Restrictions for BGP

- The BGP hold time must always be configured higher than the Graceful Restart hold time on a device, even with Graceful Restart disabled. A peer device with an unsupported hold time can establish a session with a device through an open message, but once Graceful Restart is enabled the session will flap.
- Layer 3 forwarding is delayed until routing tables are populated on a device when you switch on the device or execute the **clear ip bgp** command.

Note The routing tables require around 80 seconds for population. You can use the **show ip bgp** *ip-address* command, in privileged EXEC mode, to check whether the routing tables are populated or not.

- The maximum number of Virtual Routing and Forwarding (VRF) instances that can be configured is 16.
- When VRF-lite is configured, hosts need to communicate across BGP without multiprotocol BGP.

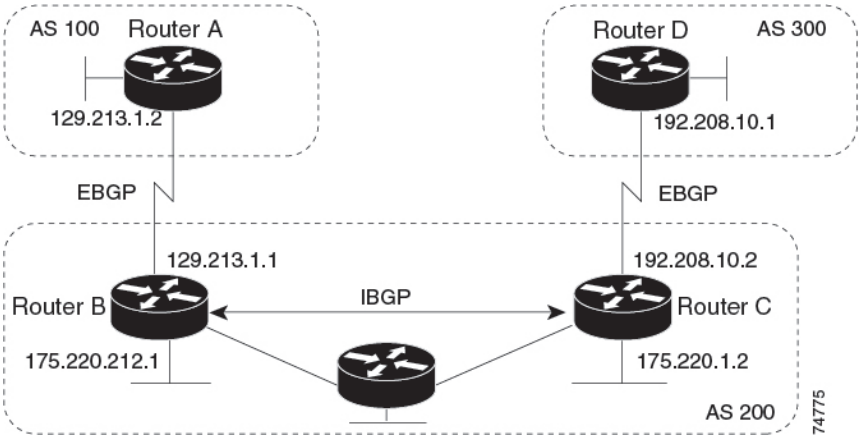
Information About BGP

The Border Gateway Protocol (BGP) is an exterior gateway protocol that is used to set up an interdomain routing system that guarantees the loop-free exchange of routing information between autonomous systems. Autonomous systems are made up of routers that operate under the same administration and that run Interior Gateway Protocols (IGPs), such as RIP or OSPF, within their boundaries and that interconnect by using an Exterior Gateway Protocol (EGP). BGP Version 4 is the standard EGP for interdomain routing in the Internet. The protocol is defined in RFCs 1163, 1267, and 1771.

BGP Network Topology

Routers that belong to the same autonomous system (AS) and that exchange BGP updates run internal BGP (IBGP), and routers that belong to different autonomous systems and that exchange BGP updates run external BGP (EBGP). Most configuration commands are the same for configuring EBGP and IBGP. The difference is that the routing updates are exchanged either between autonomous systems (EBGP) or within an AS (IBGP). The figure given below shows a network that is running both EBGP and IBGP.

Figure 45: EBGP, IBGP, and Multiple Autonomous Systems



Before exchanging information with an external AS, BGP ensures that networks within the AS can be reached by defining internal BGP peering among routers within the AS and by redistributing BGP routing information to IGPs that run within the AS, such as IGRP and OSPF.

Routers that run a BGP routing process are often referred to as BGP speakers. BGP uses the Transmission Control Protocol (TCP) as its transport protocol (specifically port 179). Two BGP speakers that have a TCP connection to each other for exchanging routing information are known as peers or neighbors. In the above figure, Routers A and B are BGP peers, as are Routers B and C and Routers C and D. The routing information is a series of AS numbers that describe the full path to the destination network. BGP uses this information to construct a loop-free map of autonomous systems.

The network has these characteristics:

- Routers A and B are running EBGP, and Routers B and C are running IBGP. Note that the EBGP peers are directly connected and that the IBGP peers are not. As long as there is an IGP running that allows the two neighbors to reach one another, IBGP peers do not have to be directly connected.
- All BGP speakers within an AS must establish a peer relationship with each other. That is, the BGP speakers within an AS must be fully meshed logically. BGP4 provides two techniques that reduce the requirement for a logical full mesh: confederations and route reflectors.
- AS 200 is a transit AS for AS 100 and AS 300—that is, AS 200 is used to transfer packets between AS 100 and AS 300.

BGP peers initially exchange their full BGP routing tables and then send only incremental updates. BGP peers also exchange keepalive messages (to ensure that the connection is up) and notification messages (in response to errors or special conditions).

In BGP, each route consists of a network number, a list of autonomous systems that information has passed through (the autonomous system path), and a list of other path attributes. The primary function of a BGP

system is to exchange network reachability information, including information about the list of AS paths, with other BGP systems. This information can be used to determine AS connectivity, to prune routing loops, and to enforce AS-level policy decisions.

A router or device running Cisco IOS does not select or use an IBGP route unless it has a route available to the next-hop router and it has received synchronization from an IGP (unless IGP synchronization is disabled). When multiple routes are available, BGP bases its path selection on attribute values. See the "Configuring BGP Decision Attributes" section for information about BGP attributes.

BGP Version 4 supports classless interdomain routing (CIDR) so you can reduce the size of your routing tables by creating aggregate routes, resulting in supernets. CIDR eliminates the concept of network classes within BGP and supports the advertising of IP prefixes.

Information About BGP Routing

To enable BGP routing, you establish a BGP routing process and define the local network. Because BGP must completely recognize the relationships with its neighbors, you must also specify a BGP neighbor.

BGP supports two kinds of neighbors: internal and external. Internal neighbors are in the same AS; external neighbors are in different autonomous systems. External neighbors are usually adjacent to each other and share a subnet, but internal neighbors can be anywhere in the same AS.

The switch supports the use of private AS numbers, usually assigned by service providers and given to systems whose routes are not advertised to external neighbors. The private AS numbers are from 64512 to 65535. You can configure external neighbors to remove private AS numbers from the AS path by using the **neighbor remove-private-as** router configuration command. Then when an update is passed to an external neighbor, if the AS path includes private AS numbers, these numbers are dropped.

If your AS will be passing traffic through it from another AS to a third AS, it is important to be consistent about the routes it advertises. If BGP advertised a route before all routers in the network had learned about the route through the IGP, the AS might receive traffic that some routers could not yet route. To prevent this from happening, BGP must wait until the IGP has propagated information across the AS so that BGP is synchronized with the IGP. Synchronization is enabled by default. If your AS does not pass traffic from one AS to another AS, or if all routers in your autonomous systems are running BGP, you can disable synchronization, which allows your network to carry fewer routes in the IGP and allows BGP to converge more quickly.

Routing Policy Changes

Routing policies for a peer include all the configurations that might affect inbound or outbound routing table updates. When you have defined two routers as BGP neighbors, they form a BGP connection and exchange routing information. If you later change a BGP filter, weight, distance, version, or timer, or make a similar configuration change, you must reset the BGP sessions so that the configuration changes take effect.

There are two types of reset, hard reset and soft reset. Software also supports a soft reset without any prior configuration. To use a soft reset without preconfiguration, both BGP peers must support the soft route refresh capability, which is advertised in the OPEN message sent when the peers establish a TCP session. A soft reset allows the dynamic exchange of route refresh requests and routing information between BGP routers and the subsequent re-advertisement of the respective outbound routing table.

- When soft reset generates inbound updates from a neighbor, it is called dynamic inbound soft reset.
- When soft reset sends a set of updates to a neighbor, it is called outbound soft reset.

A soft inbound reset causes the new inbound policy to take effect. A soft outbound reset causes the new local outbound policy to take effect without resetting the BGP session. As a new set of updates is sent during outbound policy reset, a new inbound policy can also take effect.

The table that is given below lists the advantages and disadvantages hard reset and soft reset.

Table 48: Advantages and Disadvantages of Hard and Soft Resets

Type of Reset	Advantages	Disadvantages
Hard reset	No memory overhead	The prefixes in the BGP, IP, and FIB tables provided by the neighbor are lost. Not recommended.
Outbound soft reset	No configuration, no storing of routing table updates	Does not reset inbound routing table updates.
Dynamic inbound soft reset	Does not clear the BGP session and cache Does not require storing of routing table updates and has no memory overhead	Both BGP routers must support the route refresh capability.

BGP Decision Attributes

When a BGP speaker receives updates from multiple autonomous systems that describe different paths to the same destination, it must choose the single best path for reaching that destination. When chosen, the selected path is entered into the BGP routing table and propagated to its neighbors. The decision is based on the value of attributes that the update contains and other BGP-configurable factors.

When a BGP peer learns two EBGP paths for a prefix from a neighboring AS, it chooses the best path and inserts that path in the IP routing table. If BGP multipath support is enabled and the EBGP paths are learned from the same neighboring autonomous systems, instead of a single best path, multiple paths are installed in the IP routing table. Then, during packet switching, per-packet or per-destination load-balancing is performed among the multiple paths. The **maximum-paths** router configuration command controls the number of paths allowed.

These factors summarize the order in which BGP evaluates the attributes for choosing the best path:

1. If the path specifies a next hop that is inaccessible, drop the update. The BGP next-hop attribute, automatically determined by the software, is the IP address of the next hop that is going to be used to reach a destination. For EBGP, this is usually the IP address of the neighbor that is specified by the **neighbor remote-as router** configuration command. You can disable next-hop processing by using route maps or the **neighbor next-hop-self** router configuration command.
2. Prefer the path with the largest weight (a Cisco proprietary parameter). The weight attribute is local to the router and not propagated in routing updates. By default, the weight attribute is 32768 for paths that the router originates and zero for other paths. Routes with the largest weight are preferred. You can use access lists, route maps, or the **neighbor weight** router configuration command to set weights.
3. Prefer the route with the highest local preference. Local preference is part of the routing update and exchanged among routers in the same AS. The default value of the local preference attribute is 100.

You can set local preference by using the **bgp default local-preference** router configuration command or by using a route map.

4. Prefer the route that was originated by BGP running on the local router.
5. Prefer the route with the shortest AS path.
6. Prefer the route with the lowest origin type. An interior route or IGP is lower than a route learned by EGP, and an EGP-learned route is lower than one of unknown origin or learned in another way.
7. Prefer the route with the lowest multi-exit discriminator (MED) metric attribute if the neighboring AS is the same for all routes considered. You can configure the MED by using route maps or by using the **default-metric** router configuration command. When an update is sent to an IBGP peer, the MED is included.
8. Prefer the external (EBGP) path over the internal (IBGP) path.
9. Prefer the route that can be reached through the closest IGP neighbor (the lowest IGP metric). This means that the router will prefer the shortest internal path within the AS to reach the destination (the shortest path to the BGP next-hop).
10. If the following conditions are all true, insert the route for this path into the IP routing table:
    - Both the best route and this route are external.
    - Both the best route and this route are from the same neighboring autonomous system.
    - Maximum-paths is enabled.
11. If multipath is not enabled, prefer the route with the lowest IP address value for the BGP router ID. The router ID is usually the highest IP address on the router or the loopback (virtual) address, but might be implementation-specific.

Route Maps

Within BGP, route maps can be used to control and to modify routing information and to define the conditions by which routes are redistributed between routing domains. Each route map has a name that identifies the route map (*map tag*) and an optional sequence number.

BGP Filtering

You can filter BGP advertisements by using AS-path filters, such as the **as-path access-list** global configuration command and the **neighbor filter-list** router configuration command. You can also use access lists with the **neighbor distribute-list** router configuration command. Distribute-list filters are applied to network numbers. See the “Controlling Advertising and Processing in Routing Updates” section for information about the **distribute-list** command.

You can use route maps on a per-neighbor basis to filter updates and to modify various attributes. A route map can be applied to either inbound or outbound updates. Only the routes that pass the route map are sent or accepted in updates. On both inbound and outbound updates, matching is supported based on AS path, community, and network numbers. Autonomous system path matching requires the **match as-path access-list** route-map command, community based matching requires the **match community-list** route-map command, and network-based matching requires the **ip access-list** global configuration command.

Prefix List for BGP Filtering

You can use prefix lists as an alternative to access lists in many BGP route filtering commands, including the **neighbor distribute-list** router configuration command. The advantages of using prefix lists include performance improvements in loading and lookup of large lists, incremental update support, easier CLI configuration, and greater flexibility.

Filtering by a prefix list involves matching the prefixes of routes with those listed in the prefix list, as when matching access lists. When there is a match, the route is used. Whether a prefix is permitted or denied is based upon these rules:

- An empty prefix list permits all prefixes.
- An implicit deny is assumed if a given prefix does not match any entries in a prefix list.
- When multiple entries of a prefix list match a given prefix, the sequence number of a prefix list entry identifies the entry with the lowest sequence number.

By default, sequence numbers are generated automatically and incremented in units of five. If you disable the automatic generation of sequence numbers, you must specify the sequence number for each entry. You can specify sequence values in any increment. If you specify increments of one, you cannot insert additional entries into the list; if you choose large increments, you might run out of values.

BGP Community Filtering

One way that BGP controls the distribution of routing information based on the value of the COMMUNITIES attribute. The attribute is a way to groups destinations into communities and to apply routing decisions based on the communities. This method simplifies configuration of a BGP speaker to control distribution of routing information.

A community is a group of destinations that share some common attribute. Each destination can belong to multiple communities. AS administrators can define to which communities a destination belongs. By default, all destinations belong to the general Internet community. The community is identified by the COMMUNITIES attribute, an optional, transitive, global attribute in the numerical range from 1 to 4294967200. These are some predefined, well-known communities:

- **internet**—Advertise this route to the Internet community. All routers belong to it.
- **no-export**—Do not advertise this route to EBGP peers.
- **no-advertise**—Do not advertise this route to any peer (internal or external).
- **local-as**—Do not advertise this route to peers outside the local autonomous system.

Based on the community, you can control which routing information to accept, prefer, or distribute to other neighbors. A BGP speaker can set, append, or modify the community of a route when learning, advertising, or redistributing routes. When routes are aggregated, the resulting aggregate has a COMMUNITIES attribute that contains all communities from all the initial routes.

You can use community lists to create groups of communities to use in a match clause of a route map. As with an access list, a series of community lists can be created. Statements are checked until a match is found. As soon as one statement is satisfied, the test is concluded.

BGP Neighbors and Peer Groups

Often many BGP neighbors are configured with the same update policies (that is, the same outbound route maps, distribute lists, filter lists, update source, and so on). Neighbors with the same update policies can be grouped into peer groups to simplify configuration and to make updating more efficient. When you have configured many peers, we recommend this approach.

To configure a BGP peer group, you create the peer group, assign options to the peer group, and add neighbors as peer group members. You configure the peer group by using the **neighbor** router configuration commands. By default, peer group members inherit all the configuration options of the peer group, including the remote-as (if configured), version, update-source, out-route-map, out-filter-list, out-dist-list, minimum-advertisement-interval, and next-hop-self. All peer group members also inherit changes that are made to the peer group. Members can also be configured to override the options that do not affect outbound updates.

Aggregate Routes

Classless interdomain routing (CIDR) enables you to create aggregate routes (or supernets) to minimize the size of routing tables. You can configure aggregate routes in BGP either by redistributing an aggregate route into BGP or by creating an aggregate entry in the BGP routing table. An aggregate address is added to the BGP table when there is at least one more specific entry in the BGP table.

Routing Domain Confederations

One way to reduce the IBGP mesh is to divide an autonomous system into multiple subautonomous systems and to group them into a single confederation that appears as a single autonomous system. Each autonomous system is fully meshed within itself and has a few connections to other autonomous systems in the same confederation. Even though the peers in different autonomous systems have EBGP sessions, they exchange routing information as if they were IBGP peers. Specifically, the next hop, MED, and local preference information are preserved. You can then use a single IGP for all of the autonomous systems.

BGP Route Reflectors

BGP requires that all of the IBGP speakers be fully meshed. When a router receives a route from an external neighbor, it must advertise it to all internal neighbors. To prevent a routing information loop, all IBPG speakers must be connected. The internal neighbors do not send routes that are learned from internal neighbors to other internal neighbors.

With route reflectors, all IBGP speakers need not be fully meshed because another method is used to pass learned routes to neighbors. When you configure an internal BGP peer to be a route reflector, it is responsible for passing IBGP learned routes to a set of IBGP neighbors. The internal peers of the route reflector are divided into two groups: client peers and nonclient peers (all the other routers in the autonomous system). A route reflector reflects routes between these two groups. The route reflector and its client peers form a cluster. The nonclient peers must be fully meshed with each other, but the client peers need not be fully meshed. The clients in the cluster do not communicate with IBGP speakers outside their cluster.

When the route reflector receives an advertised route, it takes one of these actions, depending on the neighbor:

- A route from an external BGP speaker is advertised to all clients and nonclient peers.
- A route from a nonclient peer is advertised to all clients.

- A route from a client is advertised to all clients and nonclient peers. Hence, the clients need not be fully meshed.

Usually a cluster of clients has a single route reflector, and the cluster is identified by the route reflector router ID. To increase redundancy and to avoid a single point of failure, a cluster might have more than one route reflector. In this case, all route reflectors in the cluster must be configured with the same 4-byte cluster ID so that a route reflector can recognize updates from route reflectors in the same cluster. All the route reflectors serving a cluster should be fully meshed and should have identical sets of client and nonclient peers.

Route Dampening

Route flap dampening is a BGP feature designed to minimize the propagation of flapping routes across an internetwork. A route is considered to be flapping when it is repeatedly available, then unavailable, then available, then unavailable, and so on. When route dampening is enabled, a numeric penalty value is assigned to a route when it flaps. When a route's accumulated penalties reach a configurable limit, BGP suppresses advertisements of the route, even if the route is running. The reuse limit is a configurable value that is compared with the penalty. If the penalty is less than the reuse limit, a suppressed route that is up is advertised again.

Dampening is not applied to routes that are learned by IBGP. This policy prevents the IBGP peers from having a higher penalty for routes external to the AS.

Conditional BGP Route Injection

Routes that are advertised through the BGP are commonly aggregated to minimize the number of routes that are used and reduce the size of global routing tables. However, common route aggregation can obscure more specific routing information that is more accurate but not necessary to forward packets to their destinations. Routing accuracy is obscured by common route aggregation because a prefix that represents multiple addresses or hosts over a large topological area cannot be accurately reflected in a single route. Cisco software provides several methods by which you can originate a prefix into BGP. Prior to the BGP conditional route injection feature, the existing methods included redistribution and using the **network** or **aggregate-address** command. However, these methods assume the existence of more specific routing information (matching the route to be originated) in either the routing table or the BGP table.

BGP conditional route injection allows you to originate a prefix into a BGP routing table without the corresponding match. This feature allows more specific routes to be generated based on administrative policy or traffic engineering information in order to provide more specific control over the forwarding of packets to these more specific routes, which are injected into the BGP routing table only if the configured conditions are met. Enabling this feature will allow you to improve the accuracy of common route aggregation by conditionally injecting or replacing less specific prefixes with more specific prefixes. Only prefixes that are equal to or more specific than the original prefix may be injected. BGP conditional route injection is enabled with the **bgp inject-map exist-map**command and uses two route maps (inject map and exist map) to install one (or more) more specific prefixes into a BGP routing table. The exist map specifies the prefixes that the BGP speaker will track. The inject map defines the prefixes that will be created and installed into the local BGP table.

Note Inject maps and exist maps will only match a single prefix per route map clause. To inject additional prefixes, you must configure additional route map clauses. If multiple prefixes are used, the first prefix that is matched will be used.

BGP Peer Templates

To address some of the limitations of peer groups such as configuration management, BGP peer templates were introduced to support the BGP update group configuration.

A peer template is a configuration pattern that can be applied to neighbors that share policies. Peer templates are reusable and support inheritance, which allows the network operator to group and apply distinct neighbor configurations for BGP neighbors that share policies. Peer templates also allow the network operator to define complex configuration patterns through the capability of a peer template to inherit a configuration from another peer template.

There are two types of peer templates:

- Peer session templates are used to group and apply the configuration of general session commands that are common to all address family and NLRI configuration modes.
- Peer policy templates are used to group and apply the configuration of commands that are applied within specific address families and NLRI configuration modes.

Peer templates improve the flexibility and enhance the capability of neighbor configuration. Peer templates also provide an alternative to peer group configuration and overcome some limitations of peer groups. BGP peer devices using peer templates also benefit from automatic update group configuration. With the configuration of the BGP peer templates and the support of the BGP dynamic update peer groups, the network operator no longer must configure peer groups in BGP and the network can benefit from improved configuration flexibility and faster convergence.

Note A BGP neighbor cannot be configured to work with both peer groups and peer templates. A BGP neighbor can be configured to belong only to a peer group or to inherit policies from peer templates.

The following restrictions apply to the peer policy templates:

- A peer policy template can directly or indirectly inherit up to eight peer policy templates.
- A BGP neighbor cannot be configured to work with both peer groups and peer templates. A BGP neighbor can be configured to belong only to a peer group or to inherit policies only from peer templates.

Inheritance in Peer Templates

The inheritance capability is a key component of peer template operation. Inheritance in a peer template is similar to node and tree structures that are commonly found in general computing, for example, file and directory trees. A peer template can directly or indirectly inherit the configuration from another peer template. The directly inherited peer template represents the tree in the structure. The indirectly inherited peer template represents a node in the tree. Because each node also supports inheritance, branches can be created that apply the configurations of all indirectly inherited peer templates within a chain back to the directly inherited peer template or the source of the tree.

This structure eliminates the need to repeat configuration statements that are commonly reapplied to groups of neighbors because common configuration statements can be applied once and then indirectly inherited by peer templates that are applied to neighbor groups with common configurations. Configuration statements that are duplicated separately within a node and a tree are filtered out at the source of the tree by the directly

inherited template. A directly inherited template overwrites any indirectly inherited statements that are duplicated in the directly inherited template.

Inheritance expands the scalability and flexibility of neighbor configuration by allowing you to chain together peer templates configurations to create simple configurations that inherit common configuration statements or complex configurations that apply specific configuration statements along with common inherited configurations. Specific details about configuring inheritance in peer session templates and peer policy templates are provided in the following sections.

When BGP neighbors use inherited peer templates, it can be difficult to determine which policies are associated with a specific template. The **detail** keyword of the **show ip bgp template peer-policy** command displays the detailed configuration of local and inherited policies that are associated with a specific template.

Peer Session Templates

Peer session templates are used to group and apply the configuration of general session commands to groups of neighbors that share session configuration elements. General session commands that are common for neighbors that are configured in different address families can be configured within the same peer session template. Peer session templates are created and configured in peer session configuration mode. Only general session commands can be configured in a peer session template. The following general session commands are supported by peer session templates:

- **description**
- **disable-connected-check**
- **ebgp-multihop**
- **exit peer-session**
- **inherit peer-session**
- **local-as**
- **password**
- **remote-as**
- **shutdown**
- **timers**
- **translate-update**
- **update-source**
- **version**

General session commands can be configured once in a peer session template and then applied to many neighbors through the direct application of a peer session template or through indirect inheritance from a peer session template. The configuration of peer session templates simplifies the configuration of general session commands that are commonly applied to all neighbors within an autonomous system.

Peer session templates support direct and indirect inheritance. A peer can be configured with only one peer session template at a time, and that peer session template can contain only one indirectly inherited peer session template.

Note If you attempt to configure more than one inherit statement with a single peer session template, an error message will be displayed.

This behavior allows a BGP neighbor to directly inherit only one session template and indirectly inherit up to seven additional peer session templates. This allows you to apply up to a maximum of eight peer session configurations to a neighbor: the configuration from the directly inherited peer session template and the configurations from up to seven indirectly inherited peer session templates. Inherited peer session configurations are evaluated first and applied starting with the last node in the branch and ending with the directly applied peer session template configuration at the source of the tree. The directly applied peer session template will have priority over inherited peer session template configurations. Any configuration statements that are duplicated in inherited peer session templates will be overwritten by the directly applied peer session template. So, if a general session command is reapplied with a different value, the subsequent value will have priority and overwrite the previous value that was configured in the indirectly inherited template. The following examples illustrate the use of this feature.

In the following example, the general session command **remote-as 1** is applied in the peer session template named SESSION-TEMPLATE-ONE:

```
template peer-session SESSION-TEMPLATE-ONE
 remote-as 1
 exit peer-session
```

Peer session templates support only general session commands. BGP policy configuration commands that are configured only for a specific address family or NLRI configuration mode are configured with peer policy templates.

Peer Policy Templates

Peer policy templates are used to group and apply the configuration of commands that are applied within specific address families and NLRI configuration mode. Peer policy templates are created and configured in peer policy configuration mode. BGP policy commands that are configured for specific address families are configured in a peer policy template. The following BGP policy commands are supported by peer policy templates:

- **advertisement-interval**
- **allowas-in**
- **as-override**
- **capability**
- **default-originate**
- **distribute-list**
- **dmzlink-bw**
- **exit-peer-policy**
- **filter-list**
- **inherit peer-policy**

- **maximum-prefix**
- **next-hop-self**
- **next-hop-unchanged**
- **prefix-list**
- **remove-private-as**
- **route-map**
- **route-reflector-client**
- **send-community**
- **send-label**
- **soft-reconfiguration**
- **unsuppress-map**
- **weight**

Peer policy templates are used to configure BGP policy commands that are configured for neighbors that belong to specific address families. Like peer session templates, peer policy templates are configured once and then applied to many neighbors through the direct application of a peer policy template or through inheritance from peer policy templates. The configuration of peer policy templates simplifies the configuration of BGP policy commands that are applied to all neighbors within an autonomous system.

Like a peer session template, a peer policy template supports inheritance. However, there are minor differences. A directly applied peer policy template can directly or indirectly inherit configurations from up to seven peer policy templates. So, a total of eight peer policy templates can be applied to a neighbor or neighbor group. Like route maps, inherited peer policy templates are configured with sequence numbers. Also like a route map, an inherited peer policy template is evaluated starting with the **inherit peer-policy** statement with the lowest sequence number and ending with the highest sequence number. However, there is a difference; a peer policy template will not collapse like a route map. Every sequence is evaluated, and if a BGP policy command is reapplied with a different value, it will overwrite any previous value from a lower sequence number.

The directly applied peer policy template and the **inherit peer-policy** statement with the highest sequence number will always have priority and be applied last. Commands that are reapplied in subsequent peer templates will always overwrite the previous values. This behavior is designed to allow you to apply common policy configurations to large neighbor groups and specific policy configurations only to certain neighbors and neighbor groups without duplicating individual policy configuration commands.

Peer policy templates support only policy configuration commands. BGP policy configuration commands that are configured only for specific address families are configured with peer policy templates.

The configuration of peer policy templates simplifies and improves the flexibility of BGP configuration. A specific policy can be configured once and referenced many times. Because a peer policy supports up to eight levels of inheritance, very specific and very complex BGP policies can also be created.

BGP Route Map Next Hop Self

The BGP Route Map Next Hop Self feature provides a way to override the settings for bgp next-hop unchanged and bgp next-hop unchanged allpath selectively. These settings are global for an address family. For some routes this may not be appropriate. For example, static routes may need to be redistributed with a next hop of

self, but connected routes and routes learned via Interior Border Gateway Protocol (IBGP) or Exterior Border Gateway Protocol (EBGP) may continue to be redistributed with an unchanged next hop.

The BGP route map next hop self functionality modifies the existing route map infrastructure to configure a new ip next-hop self setting, which overrides the bgp next-hop unchanged and bgp next-hop unchanged allpaths settings.

The ip next-hop self setting is applicable only to VPNv4 and VPNv6 address families. Routes distributed by protocols other than BGP are not affected.

You configure a new bgp route-map priority setting to inform BGP that the route map will take priority over the settings for bgp next-hop unchanged and bgp next-hop unchanged allpath. The bgp route-map priority setting only impacts BGP. The bgp route-map priority setting has no impact unless you configure the bgp next-hop unchanged or bgp next-hop unchanged allpaths settings.

How to Configure BGP

The following sections provide configurational information about BGP.

Default BGP Configuration

The table given below shows the basic default BGP configuration.

Table 49: Default BGP Configuration

Feature	Default Setting
Aggregate address	Disabled: None defined.
AS path access list	None defined.
Auto summary	Disabled.
Best path	• The router considers *as-path* in choosing a route and does not compare similar routes from external BGP peers. • Compare router ID: Disabled.
BGP community list	• Number: None defined. When you permit a value for the community number, the list defaults to an implicit deny for everything else that has not been permitted. • Format: Cisco default format (32-bit number).
BGP confederation identifier/peers	• Identifier: None configured. • Peers: None identified.
BGP Fast external fallover	Enabled.
BGP local preference	100. The range is 0 to 4294967295 with the higher value preferred.

Feature	Default Setting
BGP network	None specified; no backdoor route advertised.
BGP route dampening	Disabled by default. When enabled: • Half-life is 15 minutes. • Re-use is 750 (10-second increments). • Suppress is 2000 (10-second increments). • Max-suppress-time is 4 times half-life; 60 minutes.
BGP router ID	The IP address of a loopback interface if one is configured or the highest IP address configured for a physical interface on the router.
Default information originate (protocol or network redistribution)	Disabled.
Default metric	Built-in, automatic metric translations.
Distance	• External route administrative distance: 20 (acceptable values are from 1 to 255). • Internal route administrative distance: 200 (acceptable values are from 1 to 255). • Local route administrative distance: 200 (acceptable values are from 1 to 255).
Distribute list	• In (filter networks received in updates): Disabled. • Out (suppress networks from being advertised in updates): Disabled.
Internal route redistribution	Disabled.
IP prefix list	None defined.
Multi exit discriminator (MED)	• Always compare: Disabled. Does not compare MEDs for paths from neighbors in different autonomous systems. • Best path compare: Disabled. • MED missing as worst path: Disabled. • Deterministic MED comparison is disabled.

Feature	Default Setting
Neighbor	• Advertisement interval: 30 seconds for external peers; 5 seconds for internal peers. • Change logging: Enabled. • Conditional advertisement: Disabled. • Default originate: No default route is sent to the neighbor. • Description: None. • Distribute list: None defined. • External BGP multihop: Only directly connected neighbors are allowed. • Filter list: None used. • Maximum number of prefixes received: No limit. • Next hop (router as next hop for BGP neighbor): Disabled. • Password: Disabled. • Peer group: None defined; no members assigned. • Prefix list: None specified. • Remote AS (add entry to neighbor BGP table): No peers defined. • Private AS number removal: Disabled. • Route maps: None applied to a peer. • Send community attributes: None sent to neighbors. • Shutdown or soft reconfiguration: Not enabled. • Timers: keepalive: 60 seconds; holdtime: 180 seconds. • Update source: Best local address. • Version: BGP Version 4. • Weight: Routes learned through BGP peer: 0; routes sourced by the local router: 32768.
Route reflector	None configured.
Synchronization (BGP and IGP)	Disabled.
Table map update	Disabled.
Timers	Keepalive: 60 seconds; holdtime: 180 seconds.

Enabling BGP Routing

The maximum number of IPv4 routes that can be configured is 4096, and the maximum number of IPv6 routes that can be configured is 2048 (shared).

To enable BGP routing, perform the following procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip routing** **Example:** `Device(config)# ip routing`	Enables IP routing.
Step 4	**router bgp** *autonomous-system* **Example:** `Device(config)# router bgp 45000`	Enables a BGP routing process, assign it an AS number, and enter router configuration mode. The AS number can be from 1 to 65535, with 64512 to 65535 designated as private autonomous numbers.
Step 5	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router)# network 10.108.0.0`	Configures a network as local to this AS, and enter it in the BGP table.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *number* **Example:** `Device(config-router)# neighbor 10.108.1.2 remote-as 65200`	Adds an entry to the BGP neighbor table specifying that the neighbor that is identified by the IP address belongs to the specified AS. For EBGP, neighbors are usually directly connected, and the IP address is the address of the interface at the other end of the connection. For IBGP, the IP address can be the address of any of the router interfaces.

	Command or Action	Purpose
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **remove-private-as** **Example:** `Device(config-router)# neighbor 172.16.2.33 remove-private-as`	(Optional) Removes private AS numbers from the AS-path in outbound routing updates.
Step 8	**synchronization** **Example:** `Device(config-router)# synchronization`	(Optional) Enables synchronization between BGP and an IGP.
Step 9	**auto-summary** **Example:** `Device(config-router)# auto-summary`	(Optional) Enables automatic network summarization. When a subnet is redistributed from an IGP into BGP, only the network route is inserted into the BGP table.
Step 10	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.
Step 11	**show ip bgp network** *network-number* **Example:** `Device# show ip bgp network 10.108.0.0`	Verifies the configuration.
Step 12	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Managing Routing Policy Changes

To learn if a BGP peer supports the route refresh capability and to reset the BGP session:

Procedure

	Command or Action	Purpose
Step 1	**show ip bgp neighbors** **Example:** `Device# show ip bgp neighbors`	Displays whether a neighbor supports the route refresh capability. When supported, this message appears for the router: *Received route refresh capability from peer.*

	Command or Action	Purpose
Step 2	**clear ip bgp** {***** | *address* | *peer-group-name*} **Example:** `Device# clear ip bgp *`	Resets the routing table on the specified connection. • Enter an asterisk (*) to specify that all connections be reset. • Enter an IP address to specify the connection to be reset. • Enter a peer group name to reset the peer group.
Step 3	**clear ip bgp** {***** | *address* | *peer-group-name*} **soft out** **Example:** `Device# clear ip bgp * soft out`	(Optional) Performs an outbound soft reset to reset the inbound routing table on the specified connection. Use this command if route refresh is supported. • Enter an asterisk (*) to specify that all connections be reset. • Enter an IP address to specify the connection to be reset. • Enter a peer group name to reset the peer group.
Step 4	**show ip bgp** **Example:** `Device# show ip bgp`	Verifies the reset by checking information about the routing table and about BGP neighbors.
Step 5	**show ip bgp neighbors** **Example:** `Device# show ip bgp neighbors`	Verifies the reset by checking information about the routing table and about BGP neighbors.

Configuring BGP Decision Attributes

To configure BGP decision attributes, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system* **Example:** `Device(config)# router bgp 4500`	Enables a BGP routing process, assign it an AS number, and enter router configuration mode.
Step 4	**bgp best-path as-path ignore** **Example:** `Device(config-router)# bgp bestpath as-path ignore`	(Optional) Configures the router to ignore AS path length in selecting a route.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **next-hop-self** **Example:** `Device(config-router)# neighbor 10.108.1.1 next-hop-self`	(Optional) Disables next-hop processing on BGP updates to a neighbor by entering a specific IP address to be used instead of the next-hop address.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **weight** *weight* **Example:** `Device(config-router)# neighbor 172.16.12.1 weight 50`	(Optional) Assign a weight to a neighbor connection. Acceptable values are from 0 to 65535; the largest weight is the preferred route. Routes that are learned through another BGP peer have a default weight of 0; routes that are sourced by the local router have a default weight of 32768.
Step 7	**default-metric** *number* **Example:** `Device(config-router)# default-metric 300`	(Optional) Sets a MED metric to set preferred paths to external neighbors. All routes without a MED will also be set to this value. The range is 1 to 4294967295. The lowest value is the most desirable.
Step 8	**bgp bestpath med missing-as-worst** **Example:** `Device(config-router)# bgp bestpath med missing-as-worst`	(Optional) Configures the switch to consider a missing MED as having a value of infinity, making the path without a MED value the least desirable path.
Step 9	**bgp always-compare med** **Example:** `Device(config-router)# bgp always-compare-med`	(Optional) Configures the switch to compare MEDs for paths from neighbors in different autonomous systems. By default, MED comparison is only done among paths in the same AS.

	Command or Action	Purpose
Step 10	**bgp bestpath med confed** **Example:** `Device(config-router)# bgp bestpath med confed`	(Optional) Configures the switch to consider the MED in choosing a path from among those advertised by different subautonomous systems within a confederation.
Step 11	**bgp deterministic med** **Example:** `Device(config-router)# bgp deterministic med`	(Optional) Configures the switch to consider the MED variable when choosing among routes advertised by different peers in the same AS.
Step 12	**bgp default local-preference** *value* **Example:** `Device(config-router)# bgp default local-preference 200`	(Optional) Change the default local preference value. The range is 0 to 4294967295; the default value is 100. The highest local preference value is preferred.
Step 13	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 14	**show ip bgp** **Example:** `Device# show ip bgp`	Verifies the reset by checking information about the routing table and about BGP neighbors.
Step 15	**show ip bgp neighbors** **Example:** `Device# show ip bgp neighbors`	Verifies the reset by checking information about the routing table and about BGP neighbors.
Step 16	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring BGP Filtering with Route Maps

To configure BGP filtering with route maps, perform the following procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map set-peer-address permit 10`	Creates a route map, and enter route-map configuration mode.
Step 4	**set ip next-hop** *ip-address* [*...ip-address*] [**peer-address**] **Example:** `Device(config)# set ip next-hop 10.1.1.3`	(Optional) Sets a route map to disable next-hop processing • In an inbound route map, set the next hop of matching routes to be the neighbor peering address, overriding third-party next hops. • In an outbound route map of a BGP peer, set the next hop to the peering address of the local router, disabling the next-hop calculation.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show route-map** [*map-name*] **Example:** `Device# show route-map`	Displays all route maps configured or only the one specified to verify configuration.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config`	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`startup-config`	

Configuring BGP Filtering by Neighbor

The maximum number of BGP neighbor sessions that can be configured is 16.

To configure BGP filter by neighbor, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system* **Example:** `Device(config)# router bgp 109`	Enables a BGP routing process, assign it an AS number, and enter router configuration mode.
Step 4	**neighbor** {*ip-address* | *peer-group name*} **distribute-list** {**access-list-number** | *name*} {**in** | **out**} **Example:** `Device(config-router)# neighbor 172.16.4.1 distribute-list 39 in`	(Optional) Filters BGP routing updates to or from neighbors as specified in an access list. **Note** You can also use the **neighbor prefix-list** router configuration command to filter updates, but you cannot use both commands to configure the same BGP peer.
Step 5	**neighbor** {*ip-address* | *peer-group name*} **route-map** *map-tag* {**in** | **out**} **Example:** `Device(config-router)# neighbor 172.16.70.24 route-map internal-map in`	(Optional) Applies a route map to filter an incoming or outgoing route.
Step 6	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
Step 7	**show ip bgp neighbors** **Example:** `Device# show ip bgp neighbors`	Verifies the configuration.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring BGP Filtering by Access Lists and Neighbors

Another method of filtering is to specify an access list filter on both incoming and outbound updates, based on the BGP autonomous system paths. Each filter is an access list based on regular expressions. To use this method, define an autonomous system path access list, and apply it to updates to and from particular neighbors.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip as-path access-list** *access-list-number* {**permit** | **deny**} *as-regular-expressions* **Example:** `Device(config)# ip as-path access-list 1 deny _65535_`	Defines a BGP-related access list.
Step 4	**router bgp** *autonomous-system* **Example:** `Device(config)# router bgp 110`	Enters BGP router configuration mode.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *peer-group name*} **filter-list** {*access-list-number* | *name*} {**in** | **out** | **weight** *weight*} **Example:** `Device(config-router)# neighbor 172.16.1.1 filter-list 1 out`	Establishes a BGP filter based on an access list.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show ip bgp neighbors** [**paths** *regular-expression*] **Example:** `Device# show ip bgp neighbors`	Verifies the configuration.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Prefix Lists for BGP Filtering

You do not need to specify a sequence number when removing a configuration entry. **Show** commands include the sequence numbers in their output.

Before using a prefix list in a command, you must set up the prefix list.

To configure prefix list for BGP filtering, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ip prefix-list** *list-name* [**seq** *seq-value*] **deny** | **permit** *network/len* [**ge** *ge-value*] [**le** *le-value*] **Example:** `Device(config)# ip prefix-list BLUE permit 172.16.1.0/24`	Creates a prefix list with an optional sequence number to **deny** or **permit** access for matching conditions. You must enter at least one **permit** or **deny** clause. • *network/len* is the network number and length (in bits) of the network mask. • (Optional) **ge** and **le** values specify the range of the prefix length to be matched. The specified *ge-value* and *le-value* must satisfy this condition: *len* < *ge-value* < *le-value* < 32
Step 4	**ip prefix-list** *list-name* **seq** *seq-value* **deny** | **permit** *network/len* [**ge** *ge-value*] [**le** *le-value*] **Example:** `Device(config)# ip prefix-list BLUE seq 10 permit 172.24.1.0/24`	(Optional) Adds an entry to a prefix list, and assign a sequence number to the entry.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show ip prefix list** [**detail** | **summary**] *name* [*network/len*] [**seq** *seq-num*] [**longer**] [**first-match**] **Example:** `Device# show ip prefix list summary test`	Verifies the configuration by displaying information about a prefix list or prefix list entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring BGP Community Filtering

By default, no COMMUNITIES attribute is sent to a neighbor. You can specify that the COMMUNITIES attribute be sent to the neighbor at an IP address by using the **neighbor send-community** router configuration command.

To configure BGP community filter, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip community-list** *community-list-number* {**permit** | **deny**} *community-number* **Example:** `Device(config)# ip community-list 1 permit 50000:10`	Creates a community list, and assigns it a number. • The *community-list-number* is an integer from 1 to 99 that identifies one or more permit or deny groups of communities. • The *community-number* is the number that is configured by a **set community** route-map configuration command.
Step 4	**router bgp** *autonomous-system* **Example:** `Device(config)# router bgp 108`	Enters BGP router configuration mode.
Step 5	**neighbor** {*ip-address* | *peer-group name*} **send-community** **Example:** `Device(config-router)# neighbor 172.16.70.23 send-community`	Specifies that the COMMUNITIES attribute be sent to the neighbor at this IP address.
Step 6	**set comm-list** *list-num* **delete** **Example:** `Device(config-router)# set comm-list 500 delete`	(Optional) Removes communities from the community attribute of an inbound or outbound update that match a standard or extended community list that is specified by a route map.

	Command or Action	Purpose
Step 7	**exit** **Example:** `Device(config-router)# end`	Returns to global configuration mode.
Step 8	**ip bgp-community new-format** **Example:** `Device(config)# ip bgp-community new format`	(Optional) Displays and parses BGP communities in the format AA:NN. A BGP community is displayed in a two-part format 2 bytes long. The Cisco default community format is in the format NNAA. In the most recent RFC for BGP, a community takes the form AA:NN, where the first part is the AS number and the second part is a 2-byte number.
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 10	**show ip bgp community** **Example:** `Device# show ip bgp community`	Verifies the configuration.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring BGP Neighbors and Peer Groups

To assign configuration options to an individual neighbor, specify any of these router configuration commands by using the neighbor IP address. To assign the options to a peer group, specify any of the commands by using the peer group name. You can disable a BGP peer or peer group without removing all the configuration information by using the **neighbor shutdown** router configuration command.

To configure BGP neighbors and peer groups, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system*	Enters BGP router configuration mode.
Step 4	**neighbor** *peer-group-name* **peer-group**	Creates a BGP peer group.
Step 5	**neighbor** *ip-address* **peer-group** *peer-group-name*	Makes a BGP neighbor a member of the peer group.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *number*	Specifies a BGP neighbor. If a peer group is not configured with a **remote-as** *number*, use this command to create peer groups containing EBGP neighbors. The range is 1 to 65535.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **description** *text*	(Optional) Associates a description with a neighbor.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*]	(Optional) Allows a BGP speaker (the local router) to send the default route 0.0.0.0 to a neighbor for use as a default route.
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **send-community**	(Optional) Specifies that the COMMUNITIES attribute be sent to the neighbor at this IP address.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface*	(Optional) Allows internal BGP sessions to use any operational interface for TCP connections.
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **ebgp-multihop**	(Optional) Allows BGP sessions, even when the neighbor is not on a directly connected segment. The multihop session is not established if the only route to the multihop peer's address is the default route (0.0.0.0).
Step 12	**neighbor** {*ip-address* | *peer-group-name*} **local-as** *number*	(Optional) Specifies an AS number to use as the local AS. The range is 1 to 65535.
Step 13	**neighbor** {*ip-address* | *peer-group-name*} **advertisement-interval** *seconds*	(Optional) Sets the minimum interval between sending BGP routing updates.
Step 14	**neighbor** {*ip-address* | *peer-group-name*} **maximum-prefix** *maximum* [*threshold*]	(Optional) Controls how many prefixes can be received from a neighbor. The range is 1 to 4294967295. The *threshold* (optional) is the

	Command or Action	Purpose
		percentage of maximum at which a warning message is generated. The default is 75 percent.
Step 15	**neighbor** {*ip-address* | *peer-group-name*} **next-hop-self**	(Optional) Disables next-hop processing on the BGP updates to a neighbor.
Step 16	neighbor {*ip-address* | *peer-group-name*} password {0-7}*string*	(Optional) Sets MD5 authentication on a TCP connection to a BGP peer. The same password must be configured on both BGP peers, or the connection between them is not made. The encryption modes supported for the password are: • 0 - no encryption/plaintext • 7 - proprietary encryption type Type 7 is used only for storing the password in the device configuration. The actual value that gets used at the time of BGP session establishment is the MD5 hash of the plaintext password. • *string* - the password string
Step 17	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}	(Optional) Applies a route map to incoming or outgoing routes.
Step 18	**neighbor** {*ip-address* | *peer-group-name*} **send-community**	(Optional) Specifies that the COMMUNITIES attribute be sent to the neighbor at this IP address.
Step 19	**neighbor** {*ip-address* | *peer-group-name*} **timers** *keepalive holdtime*	(Optional) Sets timers for the neighbor or peer group. • The *keepalive* interval is the time within which keepalive messages are sent to peers. The range is 1 to 4294967295 seconds; the default is 60. • The *holdtime* is the interval after which a peer is declared inactive after not receiving a keepalive message from it. The range is 1 to 4294967295 seconds; the default is 180.
Step 20	**neighbor** {*ip-address* | *peer-group-name*} **weight** *weight*	(Optional) Specifies a weight for all routes from a neighbor.

	Command or Action	Purpose
Step 21	**neighbor** {*ip-address* | *peer-group-name*} **distribute-list** {*access-list-number* | *name*} {**in** | **out**}	(Optional) Filter BGP routing updates to or from neighbors, as specified in an access list.
Step 22	**neighbor** {*ip-address* | *peer-group-name*} **filter-list** *access-list-number* {**in** | **out** | **weight** *weight*}	(Optional) Establish a BGP filter.
Step 23	**neighbor** {*ip-address* | *peer-group-name*} **version** *value*	(Optional) Specifies the BGP version to use when communicating with a neighbor.
Step 24	**neighbor** {*ip-address* | *peer-group-name*} **soft-reconfiguration inbound**	(Optional) Configures the software to start storing received updates.
Step 25	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 26	**show ip bgp neighbors**	Verifies the configuration.
Step 27	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Aggregate Addresses in a Routing Table

To configure aggregate addresses in a routing table, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system* **Example:** `Device(config)# router bgp 106`	Enters BGP router configuration mode.
Step 4	**aggregate-address** *address mask* **Example:** `Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0`	Creates an aggregate entry in the BGP routing table. The aggregate route is advertised as coming from the AS, and the atomic aggregate attribute is set to indicate that information might be missing.
Step 5	**aggregate-address** *address mask* **as-set** **Example:** `Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0 as-set`	(Optional) Generates AS set path information. This command creates an aggregate entry following the same rules as the previous command, but the advertised path will be an AS_SET consisting of all elements contained in all paths. Do not use this keyword when aggregating many paths because this route must be continually withdrawn and updated.
Step 6	**aggregate-address** *address-mask* **summary-only** **Example:** `Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0 summary-only`	(Optional) Advertises summary addresses only.
Step 7	**aggregate-address** *address mask* **suppress-map** *map-name* **Example:** `Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0 suppress-map map1`	(Optional) Suppresses selected, more specific routes.
Step 8	**aggregate-address** *address mask* **advertise-map** *map-name* **Example:** `Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0 advertise-map map2`	(Optional) Generates an aggregate based on conditions that are specified by the route map.
Step 9	**aggregate-address** *address mask* **attribute-map** *map-name* **Example:** `Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0 attribute-map map3`	(Optional) Generates an aggregate with attributes that are specified in the route map.

	Command or Action	Purpose
Step 10	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 11	**show ip bgp neighbors** [**advertised-routes**] **Example:** `Device# show ip bgp neighbors`	Verifies the configuration.
Step 12	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Routing Domain Confederations

You must specify a confederation identifier that acts as the autonomous system number for the group of autonomous systems.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system* **Example:** `Device(config)# router bgp 100`	Enters BGP router configuration mode.
Step 4	**bgp confederation identifier** *autonomous-system* **Example:**	Configures a BGP confederation identifier.

	Command or Action	Purpose
	`Device(config)# bgp confederation identifier 50007`	
Step 5	**bgp confederation peers** *autonomous-system* [*autonomous-system ...*] **Example:** `Device(config)# bgp confederation peers 51000 51001 51002`	Specifies the autonomous systems that belong to the confederation and that will be treated as special EBGP peers.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show ip bgp neighbor** **Example:** `Device# show ip bgp neighbor`	Verifies the configuration.
Step 8	**show ip bgp network** **Example:** `Device# show ip bgp network`	Verifies the configuration.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring BGP Route Reflectors

To configure BGP route reflectors, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system* **Example:** `Device(config)# router bgp 101`	Enters BGP router configuration mode.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router)# neighbor 172.16.70.24 route-reflector-client`	Configures the local router as a BGP route reflector and the specified neighbor as a client.
Step 5	**bgp cluster-id** *cluster-id* **Example:** `Device(config-router)# bgp cluster-id 10.0.1.2`	(Optional) Configures the cluster ID if the cluster has more than one route reflector.
Step 6	**no bgp client-to-client reflection** **Example:** `Device(config-router)# no bgp client-to-client reflection`	(Optional) Disables client-to-client route reflection. By default, the routes from a route reflector client are reflected to other clients. However, if the clients are fully meshed, the route reflector does not need to reflect routes to clients.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show ip bgp** **Example:** `Device# show ip bgp`	Verifies the configuration. Displays the originator ID and the cluster-list attributes.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Route Dampening

To configure route dampening, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system* **Example:** `Device(config)# router bgp 100`	Enters BGP router configuration mode.
Step 4	**bgp dampening** **Example:** `Device(config-router)# bgp dampening`	Enables BGP route dampening.
Step 5	**bgp dampening** *half-life reuse suppress max-suppress* [**route-map** *map*] **Example:** `Device(config-router)# bgp dampening 30 1500 10000 120`	(Optional) Changes the default values of route dampening factors.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show ip bgp flap-statistics** [{**regexp** *regexp*} | {**filter-list** *list*} | {*address mask* [**longer-prefix**]}] **Example:** `Device# show ip bgp flap-statistics`	(Optional) Monitors the flaps of all paths that are flapping. The statistics are deleted when the route is not suppressed and is stable.

	Command or Action	Purpose
Step 8	**show ip bgp dampened-paths** **Example:** `Device# show pi bgp dampened-paths`	(Optional) Displays the dampened routes, including the time remaining before they are suppressed.
Step 9	**clear ip bgp flap-statistics** [{**regexp** *regexp*} | {**filter-list** *list*} | {*address mask* [**longer-prefix**]} **Example:** `Device# clear ip bgp flap-statistics`	(Optional) Clears BGP flap statistics to make it less likely that a route will be dampened.
Step 10	**clear ip bgp dampening** **Example:** `Device# clear ip bgp dampening`	(Optional) Clears route dampening information, and unsuppress the suppressed routes.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Conditionally Injecting BGP Routes

Use this task to inject more specific prefixes into a BGP routing table over less specific prefixes that were selected through normal route aggregation. These more specific prefixes can be used to provide a finer granularity of traffic engineering or administrative control than is possible with aggregated routes.

To conditionally injecting BGp routes, perform this procedure:

Before you begin

This task assumes that the IGP is already configured for the BGP peers.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.
Step 4	**bgp inject-map** *inject-map-name* **exist-map** *exist-map-name* [**copy-attributes**] **Example:** `Device(config-router)# bgp inject-map ORIGINATE exist-map LEARNED_PATH`	Specifies the inject map and the exist map for conditional route injection. • Use the **copy-attributes** keyword to specify that the injected route inherits the attributes of the aggregate route.
Step 5	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 6	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map LEARNED_PATH permit 10`	Configures a route map and enters route map configuration mode.
Step 7	**match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]} **Example:** `Device(config-route-map)# match ip address prefix-list SOURCE`	Specifies the aggregate route to which a more specific route will be injected. • In this example, the prefix list that is named SOURCE is used to redistribute the source of the route.
Step 8	**match ip route-source** {*access-list-number* | *access-list-name*} [*access-list-number...*| *access-list-name...*] **Example:** `Device(config-route-map)# match ip route-source prefix-list ROUTE_SOURCE`	Specifies the match conditions for redistributing the source of the route. • In this example, the prefix list that is named ROUTE_SOURCE is used to redistribute the source of the route. **Note** The route source is the neighbor address that is configured with the **neighbor remote-as** command. The tracked prefix must come from this neighbor in order for conditional route injection to occur.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 10	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map ORIGINATE permit 10`	Configures a route map and enters route map configuration mode.
Step 11	**set ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]} **Example:** `Device(config-route-map)# set ip address prefix-list ORIGINATED_ROUTES`	Specifies the routes to be injected. In this example, the prefix list that is named originated_routes is used to redistribute the source of the route.
Step 12	**set community** {*community-number* [**additive**] [*well-known-community*] | **none**} **Example:** `Device(config-route-map)# set community 14616:555 additive`	Sets the BGP community attribute of the injected route.
Step 13	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 14	**ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network/length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*] **Example:** `Device(config)# ip prefix-list SOURCE permit 10.1.1.0/24`	Configures a prefix list. In this example, the prefix list that is named SOURCE is configured to permit routes from network 10.1.1.0/24.
Step 15	Repeat Step 14 for every prefix list to be created.	--
Step 16	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 17	**show ip bgp injected-paths** **Example:** `Device# show ip bgp injected-paths`	(Optional) Displays information about injected paths.

Configuring Peer Session Templates

Use the following tasks to create and configure a peer session template:

Configuring a Basic Peer Session Template

Perform this task to create a basic peer session template with general BGP routing session commands that can be applied to many neighbors using one of the next two tasks.

Note The commands in Step 5 and 6 are optional and could be replaced with any supported general session commands.

Note The following restrictions apply to the peer session templates:

- A peer session template can directly inherit only one session template, and each inherited session template can also contain one indirectly inherited session template. So, a neighbor or neighbor group can be configured with only one directly applied peer session template and seven additional indirectly inherited peer session templates.
- A BGP neighbor cannot be configured to work with both peer groups and peer templates. A BGP neighbor can be configured to belong only to a peer group or to inherit policies only from peer templates.

To configure a basic peer session template, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 101`	Enters router configuration mode and creates a BGP routing process.
Step 4	**template peer-session** *session-template-name* **Example:** `Device(config-router)# template peer-session INTERNAL-BGP`	Enters session-template configuration mode and creates a peer session template.
Step 5	**remote-as** *autonomous-system-number* **Example:** `Device(config-router-stmp)# remote-as 202`	(Optional) Configures peering with a remote neighbor in the specified autonomous system. **Note** Any supported general session command can be used here. For a list of the supported commands, see the "Restrictions" section.
Step 6	**timers** *keepalive-interval hold-time* **Example:** `Device(config-router-stmp)# timers 30 300`	(Optional) Configures BGP keepalive and hold timers. The hold time must be at least twice the keepalive time. **Note** Any supported general session command can be used here. For a list of the supported commands, see the "Restrictions" section.
Step 7	**end** **Example:** `Device(config-router)# end`	Exits session-template configuration mode and returns to privileged EXEC mode.
Step 8	**show ip bgp template peer-session** [*session-template-name*] **Example:** `Device# show ip bgp template peer-session`	Displays locally configured peer session templates. The output can be filtered to display a single peer policy template with the *session-template-name* argument. This command also supports all standard output modifiers.

Configuring Peer Session Template Inheritance with the inherit peer-session Command

This task configures peer session template inheritance with the **inherit peer-session** command. It creates and configures a peer session template and allows it to inherit a configuration from another peer session template.

Note The commands in Steps 5 and 6 are optional and could be replaced with any supported general session commands.

To configure peer session template inheritance, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 101`	Enters router configuration mode and creates a BGP routing process.
Step 4	**template peer-session** *session-template-name* **Example:** `Device(config-router)# template peer-session CORE1`	Enter session-template configuration mode and creates a peer session template.
Step 5	**description** *text-string* **Example:** `Device(config-router-stmp)# description CORE-123`	(Optional) Configures a description. The text string can be up to 80 characters. **Note** Any supported general session command can be used here. For a list of the supported commands, see the "Restrictions" section.
Step 6	**update-source** *interface-type interface-number* **Example:** `Device(config-router-stmp)# update-source loopback 1`	(Optional) Configures a router to select a specific source or interface to receive routing table updates. The example uses a loopback interface. The advantage to this configuration is that the loopback interface is not as susceptible to the effects of a flapping interface. **Note**

	Command or Action	Purpose
		Any supported general session command can be used here. For a list of the supported commands, see the "Restrictions" section.
Step 7	**inherit peer-session** *session-template-name* **Example:** `Device(config-router-stmp)# inherit peer-session INTERNAL-BGP`	Configures this peer session template to inherit the configuration of another peer session template. The example configures this peer session template to inherit the configuration from INTERNAL-BGP. This template can be applied to a neighbor, and the configuration INTERNAL-BGP will be applied indirectly. No additional peer session templates can be directly applied. However, the directly inherited template can contain up to seven indirectly inherited peer session templates.
Step 8	**end** **Example:** `Device(config-router)# end`	Exits session-template configuration mode and enters privileged EXEC mode.
Step 9	**show ip bgp template peer-session** [*session-template-name*] **Example:** `Device# show ip bgp template peer-session`	Displays locally configured peer session templates. The output can be filtered to display a single peer policy template with the optional *session-template-name* argument. This command also supports all standard output modifiers.

Configuring Peer Session Template Inheritance with the neighbor inherit peer-session Command

This task configures a device to send a peer session template to a neighbor to inherit the configuration from the specified peer session template with the **neighbor inherit peer-session** command. Use the following steps to send a peer session template configuration to a neighbor to inherit.

To configure peer session template inheritance, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 101`	Enters router configuration mode and creates a BGP routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 172.16.0.1 remote-as 202`	Configures a peering session with the specified neighbor. The explicit **remote-as** statement is required for the neighbor inherit statement in Step 5 to work. If a peering is not configured, the specified neighbor in Step 5 will not accept the session template.
Step 5	**neighbor** *ip-address* **inherit peer-session** *session-template-name* **Example:** `Device(config-router)# neighbor 172.16.0.1 inherit peer-session CORE1`	Sends a peer session template to a neighbor so that the neighbor can inherit the configuration. The example configures a device to send the peer session template named CORE1 to the 172.16.0.1 neighbor to inherit. This template can be applied to a neighbor, and if another peer session template is indirectly inherited in CORE1, the indirectly inherited configuration will also be applied. No additional peer session templates can be directly applied. However, the directly inherited template can also inherit up to seven additional indirectly inherited peer session templates.
Step 6	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 7	**show ip bgp template peer-session** [*session-template-name*] **Example:** `Device# show ip bgp template peer-session`	Displays locally configured peer session templates. The output can be filtered to display a single peer policy template with the optional *session-template-name* argument. This command also supports all standard output modifiers.

Configuring Peer Policy Templates

Use the following tasks to create and configure a peer policy template:

Configuring Basic Peer Policy Templates

Perform this task to create a basic peer policy template with BGP policy configuration commands that can be applied to many neighbors using one of the next two tasks.

Note The commands in Steps 5 through 7 are optional and could be replaced with any supported BGP policy configuration commands.

Note The following restrictions apply to the peer policy templates:

- A peer policy template can directly or indirectly inherit up to eight peer policy templates.
- A BGP neighbor cannot be configured to work with both peer groups and peer templates. A BGP neighbor can be configured to belong only to a peer group or to inherit policies only from peer templates.

To configure basic peer policy templates, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**template peer-policy** *policy-template-name* **Example:** `Device(config-router)# template peer-policy GLOBAL`	Enters policy-template configuration mode and creates a peer policy template.

	Command or Action	Purpose
Step 5	**maximum-prefix** *prefix-limit* [*threshold*] [**restart** *restart-interval* | **warning-only**] **Example:** `Device(config-router-ptmp)# maximum-prefix 10000`	(Optional) Configures the maximum number of prefixes that a neighbor accept from this peer. **Note** Any supported BGP policy configuration command can be used here. For a list of the supported commands, see the "Peer Policy Templates" section.
Step 6	**weight** *weight-value* **Example:** `Device(config-router-ptmp)# weight 300`	(Optional) Sets the default weight for routes that are sent from this neighbor. **Note** Any supported BGP policy configuration command can be used here. For a list of the supported commands, see the "Peer Policy Templates" section.
Step 7	**prefix-list** *prefix-list-name* {**in** | **out**} **Example:** `Device(config-router-ptmp)# prefix-list NO-MARKETING in`	(Optional) Filters prefixes that are received by the router or sent from the router. The prefix list in the example filters inbound internal addresses. **Note** Any supported BGP policy configuration command can be used here. For a list of the supported commands, see the "Peer Policy Templates" section.
Step 8	**end** **Example:** `Device(config-router-ptmp)# end`	Exits policy-template configuration mode and returns to privileged EXEC mode.

Configuring Peer Policy Template Inheritance with the inherit peer-policy Command

This task configures peer policy template inheritance using the **inherit peer-policy**command. It creates and configure a peer policy template and allows it to inherit a configuration from another peer policy template.

Note The commands in Steps 5 and 6 are optional and could be replaced with any supported BGP policy configuration commands.

To configure peer policy template inheritance, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**template peer-policy** *policy-template-name* **Example:** `Device(config-router)# template peer-policy NETWORK1`	Enter policy-template configuration mode and creates a peer policy template.
Step 5	**route-map** *map-name* {**in**| **out**} **Example:** `Device(config-router-ptmp)# route-map ROUTE in`	(Optional) Applies the specified route map to inbound or outbound routes. **Note** Any supported BGP policy configuration command can be used here.
Step 6	**inherit peer-policy** *policy-template-name sequence-number* **Example:** `Device(config-router-ptmp)# inherit peer-policy GLOBAL 10`	Configures the peer policy template to inherit the configuration of another peer policy template. • The *sequence-number* argument sets the order in which the peer policy template is evaluated. Like a route map sequence number, the lowest sequence number is evaluated first. • The example configures this peer policy template to inherit the configuration from GLOBAL. If the template created in these steps is applied to a neighbor, the configuration GLOBAL will also be inherited and applied indirectly. Up to six additional peer policy templates can be indirectly inherited from GLOBAL for a

	Command or Action	Purpose
		total of eight directly applied and indirectly inherited peer policy templates. • This template in the example will be evaluated first if no other templates are configured with a lower sequence number.
Step 7	**end** **Example:** `Device(config-router-ptmp)# end`	Exits policy-template configuration mode and returns to privileged EXEC mode.
Step 8	**show ip bgp template peer-policy** [*policy-template-name*[**detail**]] **Example:** `Device# show ip bgp template peer-policy NETWORK1 detail`	Displays locally configured peer policy templates. • The output can be filtered to display a single peer policy template with the *policy-template-name* argument. This command also supports all standard output modifiers. • Use the **detail** keyword to display detailed policy information.

Examples

The following sample output of the **show ip bgp template peer-policy** command with the **detail** keyword displays details of the policy named NETWORK1. The output in this example shows that the GLOBAL template was inherited. Details of route map and prefix list configurations are also displayed.

```
Device# show ip bgp template peer-policy NETWORK1 detail
Template:NETWORK1, index:2.
Local policies:0x1, Inherited polices:0x80840
This template inherits:
  GLOBAL, index:1, seq_no:10, flags:0x1
Locally configured policies:
  route-map ROUTE in
Inherited policies:
  prefix-list NO-MARKETING in
  weight 300
  maximum-prefix 10000
 Template:NETWORK1 <detail>
Locally configured policies:
  route-map ROUTE in
route-map ROUTE, permit, sequence 10
  Match clauses:
    ip address prefix-lists: DEFAULT
ip prefix-list DEFAULT: 1 entries
   seq 5 permit 10.1.1.0/24
  Set clauses:
  Policy routing matches: 0 packets, 0 bytes
Inherited policies:
```

```
  prefix-list NO-MARKETING in
ip prefix-list NO-MARKETING: 1 entries
   seq 5 deny 10.2.2.0/24
```

Configuring Peer Policy Template Inheritance with the neighbor inherit peer-policy Command

This task configures a device to send a peer policy template to a neighbor to inherit using the **neighbor inherit peer-policy** command. Perform the following steps to send a peer policy template configuration to a neighbor to inherit.

When BGP neighbors use multiple levels of peer templates, it can be difficult to determine which policies are applied to the neighbor. The **policy** and **detail** keywords of the **show ip bgp neighbors** command display the inherited policies and policies that are configured directly on the specified neighbor.

To configure peer policy template, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	Configures a peering session with the specified neighbor. The explicit **remote-as** statement is required for the **neighbor inherit** statement in Step 6 to work. If a peering is not configured, the specified neighbor in Step 6 will not accept the session template.
Step 5	**address-family ipv4** [**multicast** | **unicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure a neighbor to accept address family-specific command configurations.

	Command or Action	Purpose
Step 6	**neighbor** *ip-address* **inherit peer-policy** *policy-template-name* **Example:** `Device(config-router-af)# neighbor 192.168.1.2 inherit peer-policy GLOBAL`	Sends a peer policy template to a neighbor so that the neighbor can inherit the configuration. The example configures a router to send the peer policy template that is named GLOBAL to the 192.168.1.2 neighbor to inherit. This template can be applied to a neighbor, and if another peer policy template is indirectly inherited from GLOBAL, the indirectly inherited configuration will also be applied. Up to seven additional peer policy templates can be indirectly inherited from GLOBAL.
Step 7	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 8	**show ip bgp neighbors** [*ip-address*[**policy** [**detail**]]] **Example:** `Device# show ip bgp neighbors 192.168.1.2 policy`	Displays locally configured peer policy templates. • The output can be filtered to display a single peer policy template with the *policy-template-name* argument. This command also supports all standard output modifiers. • Use the **policy** keyword to display the policies that are applied to this neighbor per address family. • Use the **detail** keyword to display detailed policy information.

Examples

The following sample output shows the policies that are applied to the neighbor at 192.168.1.2. The output displays both inherited policies and policies that are configured on the neighbor device. Inherited polices are policies that the neighbor inherits from a peer-group or a peer-policy template.

```
Device# show ip bgp neighbors 192.168.1.2 policy
Neighbor: 192.168.1.2, Address-Family: IPv4 Unicast
Locally configured policies:
 route-map ROUTE in
Inherited polices:
 prefix-list NO-MARKETING in
 route-map ROUTE in
 weight 300
 maximum-prefix 10000
```

Configuring BGP Route Map Next-hop Self

Perform this task to modify the existing route map by adding the ip next-hop self-setting and overriding the bgp next-hop unchanged and bgp next-hop unchanged all-paths settings.

To configure BGP route map next-hop self, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *map-tag* **permit** *sequence-number* **Example:** `Device(config)# route-map static-nexthop-rewrite permit 10`	Defines conditions for redistributing routes from one routing protocol to another routing protocol and enters route-map configuration mode.
Step 4	**match source-protocol** *source-protocol* **Example:** `Device(config-route-map)# match source-protocol static`	Matches Enhanced Interior Gateway Routing Protocol (EIGRP) external routes based on a source protocol.
Step 5	**set ip next-hop self** **Example:** `Device(config-route-map)# set ip next-hop self`	Configure local routes (for BGP only) with next hop of self.
Step 6	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 7	**route-map** *map-tag* **permit** *sequence-number* **Example:**	Defines conditions for redistributing routes from one routing protocol to another routing protocol and enters route-map configuration mode.

	Command or Action	Purpose
	`Device(config)# route-map static-nexthop-rewrite permit 20`	
Step 8	**match route-type internal** **Example:** `Device(config-route-map)# match route-type internal`	Redistributes routes of the specified type.
Step 9	**match route-type external** **Example:** `Device(config-route-map)# match route-type external`	Redistributes routes of the specified type.
Step 10	**match source-protocol** *source-protocol* **Example:** `Device(config-route-map)# match source-protocol connected`	Matches Enhanced Interior Gateway Routing Protocol (EIGRP) external routes based on a source protocol.
Step 11	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 12	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 13	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 172.16.232.50 remote-as 65001`	Adds an entry to the BGP or multiprotocol BGP neighbor table.
Step 14	**address-family vpnv4** **Example:** `Device(config-router)# address-family vpnv4`	Specifies the VPNv4 address family and enters address family configuration mode.
Step 15	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **activate** **Example:**	Enables the exchange of information with a Border Gateway Protocol (BGP) neighbor.

	Command or Action	Purpose
	`Device(config-router-af)# neighbor 172.16.232.50 activate`	
Step 16	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **next-hop unchanged allpaths** **Example:** `Device(config-router-af)# neighbor 172.16.232.50 next-hop unchanged allpaths`	Enables an external EBGP peer that is configured as multihop to propagate the next hop unchanged.
Step 17	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-map** *map-name* **out** **Example:** `Device(config-router-af)# neighbor 172.16.232.50 route-map static-nexthop-rewrite out`	Applies a route map to an outgoing route.
Step 18	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 19	**address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast vrf inside`	Specifies the IPv4 address family and enters address family configuration mode.
Step 20	**bgp route-map priority** **Example:** `Device(config-router-af)# bgp route-map priority`	Configures the route map priority for the local BGP routing process
Step 21	**redistribute** *protocol* **Example:** `Device(config-router-af)# redistribute static`	Redistributes routes from one routing domain into another routing domain.
Step 22	**redistribute** *protocol* **Example:**	Redistributes routes from one routing domain into another routing domain.

	Command or Action	Purpose
	`Device(config-router-af)# redistribute connected`	
Step 23	**exit-address-family** **Example:** `Device(config-router-af)# exit address-family`	Exits address family configuration mode and enters router configuration mode.
Step 24	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.

Configuration Examples for BGP

The following sections provide configuration examples for BGP.

Example: Configuring Conditional BGP Route Injection

The following sample output is similar to the output that will be displayed when the **show ip bgp injected-paths** command is entered:

```
Device# show ip bgp injected-paths

BGP table version is 11, local router ID is 10.0.0.1
Status codes:s suppressed, d damped, h history, * valid, > best, i -
internal
Origin codes:i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 172.16.0.0       10.0.0.2                               0 ?
*> 172.17.0.0/16    10.0.0.2                               0 ?
```

Example: Configuring Peer Session Templates

The following example creates a peer session template that is named INTERNAL-BGP in session-template configuration mode:

```
router bgp 45000
 template peer-session INTERNAL-BGP
 remote-as 50000
 timers 30 300
 exit-peer-session
```

The following example creates a peer session template named CORE1. This example inherits the configuration of the peer session template named INTERNAL-BGP.

```
router bgp 45000
 template peer-session CORE1
```

```
 description CORE-123
 update-source loopback 1
 inherit peer-session INTERNAL-BGP
 exit-peer-session
```

The following example configures the 192.168.3.2 neighbor to inherit the CORE1 peer session template. The 192.168.3.2 neighbor will also indirectly inherit the configuration from the peer session template named INTERNAL-BGP. The explicit **remote-as** statement is required for the neighbor inherit statement to work. If a peering is not configured, the specified neighbor will not accept the session template.

```
router bgp 45000
 neighbor 192.168.3.2 remote-as 50000
 neighbor 192.168.3.2 inherit peer-session CORE1
```

Examples: Configuring Peer Policy Templates

The following example creates a peer policy template that is named GLOBAL and enters policy-template configuration mode:

```
router bgp 45000
 template peer-policy GLOBAL
  weight 1000
  maximum-prefix 5000
  prefix-list NO_SALES in
  exit-peer-policy
```

The following example creates a peer policy template that is named PRIMARY-IN and enters policy-template configuration mode:

```
router bgp 45000
 template peer-policy PRIMARY-IN
  prefix-list ALLOW-PRIMARY-A in
  route-map SET-LOCAL in
  weight 2345
  default-originate
  exit-peer-policy
```

The following example creates a peer policy template named CUSTOMER-A. This peer policy template is configured to inherit the configuration from the peer policy templates that are named PRIMARY-IN and GLOBAL.

```
router bgp 45000
 template peer-policy CUSTOMER-A
  route-map SET-COMMUNITY in
  filter-list 20 in
  inherit peer-policy PRIMARY-IN 20
  inherit peer-policy GLOBAL 10
  exit-peer-policy
```

The following example configures the 192.168.2.2 neighbor in address family mode to inherit the peer policy template named CUSTOMER-A. Assuming this example is a continuation of the example above, because the peer policy template named CUSTOMER-A above inherited the configuration from the templates that are named PRIMARY-IN and GLOBAL, the 192.168.2.2 neighbor will also indirectly inherit the peer policy templates that are named PRIMARY-IN and GLOBAL.

```
router bgp 45000
 neighbor 192.168.2.2 remote-as 50000
```

```
 address-family ipv4 unicast
  neighbor 192.168.2.2 inherit peer-policy CUSTOMER-A
  end
```

Example: Configuring BGP Route Map next-hop self

This section contains an example of how to configure BGP Route Map next-hop self.

In this example, a route map is configured that matches the networks where you wish to override settings for bgp next-hop unchanged and bgp next-hop unchanged allpath. Subsequently, next-hop self is configured. After this, the bgp route map priority is configured for the specified address family so that the previously specified route map takes priority over the settings for bgp next-hop unchanged and bgp next-hop unchanged allpath. This configuration results in static routes being redistributed with a next hop of self, but connected routes and routes learned via IBGP or EBGP continue to be redistributed with an unchanged next hop.

```
route-map static-nexthop-rewrite permit 10
 match source-protocol static
 set ip next-hop self
route-map static-nexthop-rewrite permit 20
 match route-type internal
 match route-type external
 match source-protocol connected
!
router bgp 65000
 neighbor 172.16.232.50 remote-as 65001
 address-family vpnv4
   neighbor 172.16.232.50 activate
   neighbor 172.16.232.50 next-hop unchanged allpaths
   neighbor 172.16.232.50 route-map static-nexthop-rewrite out
 exit-address-family
 address-family ipv4 unicast vrf inside
   bgp route-map priority
   redistribute static
   redistribute connected
 exit-address-family
end
```

Monitoring and Maintaining BGP

You can remove all contents of a particular cache, table, or database. This might be necessary when the contents of the particular structure have become or are suspected to be invalid.

You can display specific statistics, such as the contents of BGP routing tables, caches, and databases. You can use the information to get resource utilization and solve network problems. You can also display information about node reachability and discover the routing path your device's packets are taking through the network.

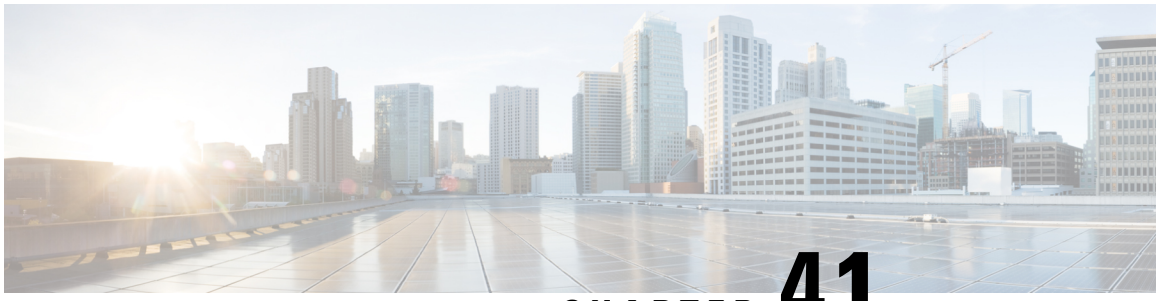
The table given below lists the privileged EXEC commands for clearing and displaying BGP.

Table 50: IP BGP Clear and Show Commands

Command	Purpose
clear ip bgp *address*	Resets a particular BGP connection.

Command	Purpose
clear ip bgp *	Resets all BGP connections.
clear ip bgp peer-group *tag*	Removes all members of a BGP peer group.
show ip bgp *prefix*	Displays peer groups and peers not in peer groups to which the prefix has been advertised. Also displays prefix attributes such as the next hop and the local prefix.
show ip bgp cidr-only	Displays all BGP routes that contain subnet and supernet network masks.
show ip bgp community [*community-number*] [**exact**]	Displays routes that belong to the specified communities.
show ip bgp community-list *community-list-number* [**exact-match**]	Displays routes that are permitted by the community list.
show ip bgp filter-list *access-list-number*	Displays routes that are matched by the specified AS path access list.
show ip bgp inconsistent-as	Displays the routes with inconsistent originating autonomous systems.
show ip bgp regexp *regular-expression*	Displays the routes that have an AS path that matches the specified regular expression entered on the command line.
show ip bgp	Displays the contents of the BGP routing table.
show ip bgp neighbors [*address*]	Displays detailed information on the BGP and TCP connections to individual neighbors.
show ip bgp neighbors [*address*] [**advertised-routes** | **dampened-routes** | **flap-statistics** | **paths** *regular-expression* | **received-routes** | **routes**]	Displays routes learned from a particular BGP neighbor.
show ip bgp paths	Displays all BGP paths in the database.
show ip bgp peer-group [*tag*] [**summary**]	Displays information about BGP peer groups.
show ip bgp summary	Displays the status of all BGP connections.

The **bgp log-neighbor changes** command is enabled by default. It allows to log messages that are generated when a BGP neighbor resets, comes up, or goes down.



CHAPTER 41

Configuring IS-IS

Information About IS-IS Routing

Integrated Intermediate System-to-Intermediate System (IS-IS) is an ISO dynamic routing protocol (described in ISO 105890). To enable IS-IS you should create an IS-IS routing process and assign it to a specific interface, rather than to a network. You can specify more than one IS-IS routing process per Layer 3 device by using the multiarea IS-IS configuration syntax. You should then configure the parameters for each instance of the IS-IS routing process.

Small IS-IS networks are built as a single area that includes all the devices in the network. As the network grows larger, the network reorganizes itself into a backbone area that is made up of all the connected set of Level 2 devices that are still connected to their local areas. Within a local area, devices know how to reach all system IDs. Between areas, devices know how to reach the backbone, and the backbone devices know how to reach other areas.

Devices establish Level 1 adjacencies to perform routing within a local area (station routing). Devices establish Level 2 adjacencies to perform routing between Level 1 areas (area routing).

A single Cisco device can participate in routing in up to 29 areas and can perform Level 2 routing in the backbone. In general, each routing process corresponds to an area. By default, the first instance of the routing process that is configured performs both Level 1 and Level 2 routing. You can configure additional device instances, which are automatically treated as Level 1 areas. You must configure the parameters for each instance of the IS-IS routing process individually.

For IS-IS multiarea routing, you can configure only one process to perform Level 2 routing, although you can define up to 29 Level 1 areas for each Cisco unit. If Level 2 routing is configured on any process, all additional processes are automatically configured as Level 1. You can configure this process to perform Level 1 routing at the same time. If Level 2 routing is not desired for a device instance, remove the Level 2 capability using the **is-type** command in global configuration mode. Use the **is-type** command also to configure a different device instance as a Level 2 device.

IS-IS Global Parameters

The following are the optional IS-IS global parameters that you can configure:

- You can force a default route into an IS-IS routing domain by configuring a default route that is controlled by a route map. You can also specify the other filtering options that are configurable under a route map.
- You can configure the device to ignore IS-IS link-state packets (LSPs) that are received with internal checksum errors, or to purge corrupted LSPs, and cause the initiator of the LSP to regenerate it.
- You can assign passwords to areas and domains.
- You can create aggregate addresses that are represented in the routing table by a summary address (based on route summarization). Routes that are learned from other routing protocols can also be summarized. The metric used to advertise the summary is the smallest metric of all the specific routes.
- You can set an overload bit.
- You can configure the LSP refresh interval and the maximum time that an LSP can remain in the device database without a refresh.
- You can set the throttling timers for LSP generation, shortest path first computation, and partial route computation.
- You can configure the device to generate a log message when an IS-IS adjacency changes state (Up or Down).
- If a link in the network has a maximum transmission unit (MTU) size of less than 1500 bytes, you can lower the LSP MTU so that routing still occurs.
- You can use the **partition avoidance** command to prevent an area from becoming partitioned when full connectivity is lost among a Level 1-2 border device, adjacent Level 1 devices, and end hosts.

IS-IS Interface Parameters

You can optionally configure certain interface-specific IS-IS parameters independently from other attached devices. However, if you change default value, such as multipliers and time intervals, it makes sense to also change them on multiple devices and interfaces. Most of the interface parameters can be configured for level 1, level 2, or both.

The following are the interface-level parameters that you can configure:

- The default metric on the interface that is used as a value for the IS-IS metric and assigned when quality of service (QoS) routing is not performed.
- The hello interval (length of time between hello packets sent on the interface) or the default hello packet multiplier used on the interface to determine the hold time sent in IS-IS hello packets. The hold time determines how long a neighbor waits for another hello packet before declaring the neighbor down. This determines how quickly a failed link or neighbor is detected so that routes can be recalculated. Change the hello multiplier in circumstances where hello packets are lost frequently and IS-IS adjacencies are failing unnecessarily. You can raise the hello multiplier and lower the hello interval correspondingly to make the hello protocol more reliable, without increasing the time required to detect a link failure.
- Other time intervals:
  - Complete sequence number PDU (CSNP) interval—CSNPs are sent by the designated device to maintain database synchronization.
  - Retransmission interval—This is the time between retransmission of IS-IS LSPs for point-to-point links.

- IS-IS LSP retransmission throttle interval—This is the maximum rate (number of milliseconds between packets) at which IS-IS LSPs are resent on point-to-point links. This interval is different from the retransmission interval, which is the time between successive retransmissions of the same LSP.

- Designated device-election priority, which allows you to reduce the number of adjacencies required on a multiaccess network, which in turn reduces the amount of routing protocol traffic and the size of the topology database.
- The interface circuit type, which is the type of adjacency required for neighbors on the specified interface.
- Password authentication for the interface.

How to Configure IS-IS

The following sections provide information on how to enable IS-IS on an interface, how to configure IS-IS global parameters, and how to configure IS-IS interface parameters.

Default IS-IS Configuration

Table 51: Default IS-IS Configuration

Feature	Default Setting
Ignore link-state PDU (LSP) errors	Enabled.
IS-IS type	Conventional IS-IS—The router acts as both a Level 1 (station) and a Level 2 (area) router. Multiarea IS-IS—The first instance of the IS-IS routing process is a Level 1-2 router. Remaining instances are Level 1 routers.
Default-information originate	Disabled.
Log IS-IS adjacency state changes.	Disabled.
LSP generation throttling timers	Maximum interval between two consecutive occurrences—5000 milliseconds. Initial LSP generation delay—50 milliseconds. Hold time between the first and second LSP generation—200 milliseconds.
LSP maximum lifetime (without a refresh)	1200 seconds (20 minutes) before the LSP packet is deleted.
LSP refresh interval	Every 900 seconds (15 minutes).
Maximum LSP packet size	1497 bytes.

Feature	Default Setting
Partial route computation (PRC) throttling timers	Maximum PRC wait interval—5000 milliseconds. Initial PRC calculation delay after a topology change—50 milliseconds. Hold time between the first and second PRC calculation—200 milliseconds.
Partition avoidance	Disabled.
Password	No area or domain password is defined, and authentication is disabled.
Set-overload-bit	Disabled. When enabled, if no arguments are entered, the overload bit is set immediately and remains set until you enter the **no set-overload-bit** command.
Shortest path first (SPF) throttling timers	Maximum interval between consecutive SFPs—5000 milliseconds. Initial SFP calculation after a topology change—200 milliseconds. Hold time between the first and second SFP calculation—50 milliseconds.
Summary-address	Disabled.

Enabling IS-IS Routing

To enable IS-IS, specify a name and a network entity title (NET) for each routing process. Enable IS-IS routing on the interface and specify the area for each instance of the routing process.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**clns routing** **Example:** `Device(config)#clns routing`	Enables ISO connectionless routing on the device.

	Command or Action	Purpose
Step 4	**router isis** [*area tag*] **Example:** `Device(config)#router isis tag1`	Enables IS-IS routing for the specified routing process and enters IS-IS routing configuration mode. (Optional) Use the *area tag* argument to identify the area to which the IS-IS router is assigned. Enter a value if you are configuring multiple IS-IS areas. The first IS-IS instance that is configured is Level 1-2 by default. Later instances are automatically configured as Level 1. You can change the level of routing by using the **is-type** command in global configuration mode.
Step 5	**net** *network-entity-title* **Example:** `Device(config-router)#net 47.0004.004d.0001.0001.0c11.1111.00`	Configures the NETs for the routing process. While configuring multiarea IS-IS, specify a NET for each routing process. Specify a name for a NET and for an address.
Step 6	**is-type** {**level-1** | **level-1-2** | **level-2-only**} **Example:** `Device(config-router)#is-type level-2-only`	(Optional) Configures the router to act as a Level 1 (station) router, a Level 2 (area) router for multiarea routing, or both (the default): • **level 1**—Acts as a station router only. • **level 1-2**—Acts as both a station router and an area router. • **level 2**—Acts as an area router only.
Step 7	**exit** **Example:** `Device(config-router)#end`	Returns to global configuration mode.
Step 8	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Specifies an interface to route IS-IS, and enters interface configuration mode. If the interface is not already configured as a Layer 3 interface, enter the **no switchport** command to configure the interface into Layer 3 mode.
Step 9	**ip router isis** [*area tag*] **Example:** `Device(config-if)#ip router isis tag1`	Configures an IS-IS routing process on the interface and attaches an area designator to the routing process.
Step 10	**ip address** *ip-address-mask* **Example:**	Defines the IP address for the interface. An IP address is required for all the interfaces in an

	Command or Action	Purpose
	`Device(config-if)#ip address 10.0.0.5 255.255.255.0`	area, that is enabled for IS-IS, if any one interface is configured for IS-IS routing.
Step 11	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 12	**show isis** [*area tag*] **database detail** **Example:** `Device#show isis database detail`	Verifies your entries.

Configuring IS-IS Global Parameters

To configure global IS-IS parameters, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**router isis** **Example:** `Device(config)#router isis`	Specifies the IS-IS routing protocol and enters router configuration mode.
Step 4	**default-information originate** [**route-map** *map-name*] **Example:** `Device(config-router)#default-information originate route-map map1`	(Optional) Forces a default route into the IS-IS routing domain. When you enter the **route-map** *map-name* command, the routing process generates the default route for a valid route map.
Step 5	**ignore-lsp-errors** **Example:**	(Optional) Configures the device to ignore LSPs with internal checksum errors, instead of purging the LSPs. This command is enabled

	Command or Action	Purpose
	`Device(config-router)#ignore-lsp-errors`	by default (corrupted LSPs are dropped). To purge the corrupted LSPs, enter the **no ignore-lsp-errors** command in router configuration mode.
Step 6	**area-password** *password* **Example:** `Device(config-router)#area-password 1password`	(Optional Configures the area authentication password that is inserted in Level 1 (station router level) LSPs.
Step 7	**domain-password** *password* **Example:** `Device(config-router)#domain-password 2password`	(Optional) Configures the routing domain authentication password that is inserted in Level 2 (area router level) LSPs.
Step 8	**summary-address** *address mask* [**level-1** | **level-1-2** | **level-2**] **Example:** `Device(config-router)#summary-address 10.1.0.0 255.255.0.0 level-2`	(Optional) Creates a summary of addresses for a given level.
Step 9	**set-overload-bit** [**on-startup** {*seconds* | **wait-for-bgp**}] **Example:** `Device(config-router)#set-overload-bit on-startup wait-for-bgp`	(Optional) Sets an overload bit to allow other devices to ignore the device in their shortest path first (SPF) calculations if the device is having problems. • (Optional) **on-startup**—Sets the overload bit only on startup. If **on-startup** is not specified, the overload bit is set immediately and remains set until you enter the **no set-overload-bit** command. If **on-startup** is specified, you must either enter number of seconds or enter **wait-for-bgp**. • *seconds*—When the **on-startup** keyword is configured, it causes the overload bit to be set when the system is started and remains set for the specified number of seconds. The range is from 5 to 86400 seconds. • **wait-for-bgp**—When the **on-startup** keyword is configured, causes the overload bit to be set when the system is started and remains set until BGP has converged. If BGP does not signal the

	Command or Action	Purpose
		IS-IS that it is converged, the IS-IS will turn off the overload bit after 10 minutes.
Step 10	**lsp-refresh-interval** *seconds* **Example:** `Device(config-router)#lsp-refresh-interval 1080`	(Optional) Sets an LSP refresh interval, in seconds. The range is from 1 to 65535 seconds. The default is to send LSP refreshes every 900 seconds (15 minutes).
Step 11	**max-lsp-lifetime** *seconds* **Example:** `Device(config-router)#max-lsp-lifetime 1000`	(Optional) Sets the maximum time that LSP packets remain in the router database without being refreshed. The range is from 1 to 65535 seconds. The default is 1200 seconds (20 minutes). After the specified time interval, the LSP packet is deleted.
Step 12	**lsp-gen-interval** [**level-1** | **level-2**] *lsp-max-wait* [*lsp-initial-wait lsp-second-wait*] **Example:** `Device(config-router)#lsp-gen-interval level-2 2 50 100`	(Optional) Sets the IS-IS LSP generation throttling timers: • *lsp-max-wait*—Maximum interval (in milliseconds) between two consecutive occurrences of an LSP being generated. The range is from 1 to 120; the default is 5000. • *lsp-initial-wait*—Initial LSP generation delay (in milliseconds). The range is from 1 to 10000; the default is 50. • *lsp-second-wait*—Hold time between the first and second LSP generation (in milliseconds). The range is from 1 to 10000; the default is 200.
Step 13	**spf-interval** [**level-1** | **level-2**] *spf-max-wait* [*spf-initial-wait spf-second-wait*] **Example:** `Device(config-router)#spf-interval level-2 5 10 20`	(Optional) Sets IS-IS SPF throttling timers. • *spf-max-wait*—Maximum interval between consecutive SFPs (in milliseconds). The range is from 1 to 120; the default is 5000. • *spf-initial-wait*—Initial SFP calculation after a topology change (in milliseconds). The range is from 1 to 10000; the default is 50. • *spf-second-wait*—Hold time between the first and second SFP calculation (in milliseconds). The range is from 1 to 10000; the default is 200.

	Command or Action	Purpose
Step 14	**prc-interval** *prc-max-wait* [*prc-initial-wait prc-second-wait*] **Example:** `Device(config-router)#prc-interval 5 10 20`	(Optional) Sets IS-IS PRC throttling timers. • *prc-max-wait*—Maximum interval (in milliseconds) between two consecutive PRC calculations. The range is from 1 to 120; the default is 5000. • *prc-initial-wait*—Initial PRC calculation delay (in milliseconds) after a topology change. The range is from 1 to 10,000; the default is 50. • *prc-second-wait*—Hold time between the first and second PRC calculation (in milliseconds). The range is from 1 to 10,000; the default is 200.
Step 15	**log-adjacency-changes** [**all**] **Example:** `Device(config-router)#log-adjacency-changes all`	(Optional) Sets the router to log IS-IS adjacency state changes. Enter **all** to include all the changes generated by events that are not related to the IS-IS hellos, including End System-to-Intermediate System PDUs and LSPs.
Step 16	**lsp-mtu** *size* **Example:** `Device(config-router)#lsp mtu 1560`	(Optional) Specifies the maximum LSP packet size, in bytes. The range is from 128 to 4352; the default is 1497 bytes. **Note** If a link in the network has a reduced MTU size, you must change the LSP MTU size on all the devices in the network.
Step 17	**partition avoidance** **Example:** `Device(config-router)#partition avoidance`	(Optional) Causes an IS-IS Level 1-2 border router to stop advertising the Level 1 area prefix into the Level 2 backbone when full connectivity is lost among the border router, all adjacent level 1 routers, and end hosts.
Step 18	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.

Configuring IS-IS Interface Parameters

To configure IS-IS interface-specific parameters, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Specifies the interface to be configured and enters interface configuration mode. If the interface is not already configured as a Layer 3 interface, enter the **no switchport** command to configure the interface into Layer 3 mode.
Step 4	**isis metric** *default-metric* [**level-1** | **level-2**] **Example:** `Device(config-if)#isis metric 15`	(Optional) Configures the metric (or cost) for the specified interface. The range is from 0 to 63; the default is 10. If no level is entered, the default is applied to both Level 1 and Level 2 routers.
Step 5	**isis hello-interval** {*seconds* | **minimal**} [**level-1** | **level-2**] **Example:** `Device(config-if)#isis hello-interval minimal`	(Optional) Specifies the length of time between the hello packets that are sent by the device. By default, a value that is three times the hello interval *seconds* is advertised as the *holdtime* in the hello packets sent. With smaller hello intervals, topological changes are detected faster, but there is more routing traffic. • **minimal**—Causes the system to compute the hello interval based on the hello multiplier so that the resulting hold time is 1 second. • *seconds*—Range is from 1 to 65535; default is 10 seconds.
Step 6	**isis hello-multiplier** *multiplier* [**level-1** | **level-2**] **Example:** `Device(config-if)#isis hello-multiplier 5`	(Optional) Specifies the number of IS-IS hello packets that a neighbor must miss before the device declares the adjacency as down. The range is from 3 to 1000; default is 3. **Note** Using a smaller hello multiplier causes fast convergence, but might result in routing instability.

	Command or Action	Purpose
Step 7	**isis csnp-interval** *seconds* [**level-1** | **level-2**] **Example:** `Device(config-if)#isis csnp-interval 15`	(Optional) Configures the IS-IS complete sequence number PDU (CSNP) interval for the interface. The range is from 0 to 65535; default is 10 seconds.
Step 8	**isis retransmit-interval** *seconds* **Example:** `Device(config-if)#isis retransmit-interval 7`	(Optional) Configures the number of seconds between the retransmission of IS-IS LSPs for point-to-point links. Specify an integer that is greater than the expected round-trip delay between any two routers on the network. The range is from 0 to 65535; default is 5 seconds.
Step 9	**isis retransmit-throttle-interval** *milliseconds* **Example:** `Device(config-if)#isis retransmit-throttle-interval 4000`	(Optional) Configures the IS-IS LSP retransmission throttle interval, which is the maximum rate (number of milliseconds between packets) at which IS-IS LSPs will be resent on point-to-point links. The range is from 0 to 65535; default is determined by the **isis lsp-interval** command.
Step 10	**isis priority** *value* [**level-1** | **level-2**] **Example:** `Device(config-if)#isis priority 50`	(Optional) Configures the priority for the designated router. The range is from 0 to 127; default is 64.
Step 11	**isis circuit-type** {**level-1** | **level-1-2** | **level-2-only**} **Example:** `Device(config-if)#isis circuit-type level-1-2`	(Optional) Configures the type of adjacency that is required for neighbors on the specified interface (specify the interface circuit type). • **level-1**—Level 1 adjacency is established if there is at least one area address that is common to both this node and its neighbors. • **level-1-2**—Level 1 and Level 2 adjacency are established if the neighbor is also configured as both Level 1 and Level 2, and there is at least one area in common. If there is no area in common, a Level 2 adjacency is established. This is the default option. • **level 2**—Level 2 adjacency is established. If the neighbor router is a Level 1 router, no adjacency is established.
Step 12	**isis password** *password* [**level-1** | **level-2**] **Example:**	(Optional) Configures the authentication password for an interface. By default, authentication is disabled. Specifying Level 1

	Command or Action	Purpose
	`Device(config-if)#isis password secret`	or Level 2 enables the password only for Level 1 or Level 2 routing, respectively. If you do not specify a level, the default is Level 1 and Level 2.
Step 13	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.

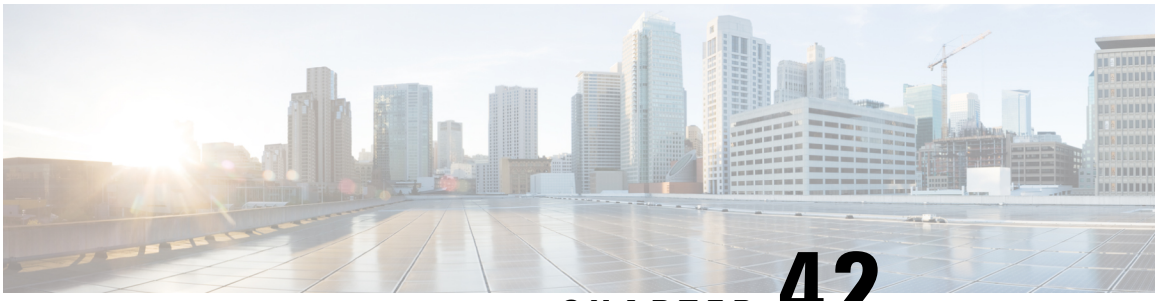
Monitoring and Maintaining IS-IS

You can display specific IS-IS statistics, such as the contents of routing tables, caches, and databases. You can also display information about specific interfaces, filters, or neighbors.

The following table lists the privileged EXEC commands for clearing and displaying IS-IS routing.

Table 52: IS-IS show Commands

Command	Purpose
show ip route isis	Displays the current state of the IS-IS IP routing table.
show isis database	Displays the IS-IS link-state database.
show isis routes	Displays the IS-IS Level 1 routing table.
show isis spf-log	Displays a history of the SPF calculations for IS-IS.
show isis topology	Displays a list of all the connected routers in all the areas.
show route-map	Displays all the route maps configured or only the one that is specified.
trace clns *destination*	Traces the paths taken to a specified destination by packets in the network.



CHAPTER 42

Configuring VRF-lite

Information About VRF-lite

VRF-lite is a feature that enables a service provider to support two or more VPNs, where IP addresses can be overlapped among the VPNs. VRF-lite uses input interfaces to distinguish routes for different VPNs and forms virtual packet-forwarding tables by associating one or more Layer 3 interfaces with each VRF. Interfaces in a VRF can be either physical, such as Ethernet ports, or logical, such as VLAN SVIs, but a Layer 3 interface cannot belong to more than one VRF at any time.

Note VRF-lite interfaces must be Layer 3 interfaces.

VRF-lite includes these devices:

- Customer edge (CE) devices provide customer access to the service provider network over a data link to one or more provider edge routers. The CE device advertises the site's local routes to the provider edge router and learns the remote VPN routes from it. A switch can be a CE.
- Provider edge (PE) routers exchange routing information with CE devices by using static routing or a routing protocol such as BGP, RIPv1, or RIPv2.

  The PE is only required to maintain VPN routes for those VPNs to which it is directly attached, eliminating the need for the PE to maintain all of the service provider VPN routes. Each PE router maintains a VRF for each of its directly connected sites. Multiple interfaces on a PE router can be associated with a single VRF if all of these sites participate in the same VPN. Each VPN is mapped to a specified VRF. After learning local VPN routes from CEs, a PE router exchanges VPN routing information with other PE routers by using internal BGP (iBGP).
- Provider routers (or core routers) are any routers in the service provider network that do not attach to CE devices.

With VRF-lite, multiple customers can share one CE. The shared CE maintains separate VRF tables for each customer and switches or routes packets for each customer based on its own routing table. VRF-lite allows a CE device to maintain separate VRF tables to extend the privacy and security of a VPN to the branch office.

To configure VRF, create a VRF table and specify the Layer 3 interface associated with the VRF.

Guidelines for Configuring VRF-lite

IPv4 and IPv6

- A switch with VRF-lite is shared by multiple customers, and all customers have their own routing tables.
- Because customers use different VRF tables, you can reuse the same IP addresses.
- VRF-lite lets multiple customers share the same physical link between the PE and the CE.
- The switch supports configuring VRF by using physical ports, VLAN SVIs, or a combination of both. You can connect SVIs through an access port or a trunk port.
- A customer can use multiple VLANs as long because they do not overlap with those of other customers. A customer's VLANs are mapped to a specific routing table ID that is used to identify the appropriate routing tables stored on the switch.
- The Layer 3 TCAM resource is shared between all VRFs. To ensure that any one VRF has sufficient CAM space, use the **maximum routes** command.
- A switch using VRF can support one global network and multiple VRFs. The total number of routes supported is limited by the size of the TCAM.
- A single VRF can be configured for both IPv4 and IPv6.
- If an incoming packet's destination address is not found in the vrf table, the packet is dropped. Also, if insufficient TCAM space exists for a VRF route, hardware switching for that VRF is disabled and the corresponding data packets are sent to software for processing.

IPv4 Specific

- The switch supports PIM-SM and PIM-SSM protocols.

IPv6 specific

- VRF-aware OSPFv3, EIGRPv6, and IPv6 static routing are supported.
- VRF-aware IPv6 route applications include: ping, telnet, ssh, tftp, ftp and traceroute. (This list does not include the management interface, which is handled differently even though you can configure both IPv4 or IPv6 VRF under it.)

How to Configure VRF-lite

This section provides information about configuring VRF-lite.

Configuring VRF-lite for IPv4

This section provides information about configuring VRF-lite for IPv4.

Configuring VRF-Aware Services

IP services can be configured on global interfaces and within the global routing instance. IP services are enhanced to run on multiple routing instances; they are VRF-aware. Any configured VRF in the system can be specified for a VRF-aware service.

VRF-aware services are implemented in platform-independent modules. VRF provides multiple routing instances in Cisco IOS. Each platform has its own limit on the number of VRFs it supports.

VRF-aware services have the following characteristics:

- The user can ping a host in a user-specified VRF.
- ARP entries are learned in separate VRFs. The user can display Address Resolution Protocol (ARP) entries for specific VRFs.

Configuring the User Interface for ARP

Procedure

	Command or Action	Purpose
Step 1	**show ip arp vrf** *vrf-name* **Example:** `Device# show ip arp vrf vrf-name`	Displays the ARP table (static and dynamic entries) in the specified VRF.
Step 2	**arp vrf** *vrf-name ip-address mac-address ARPA* **Example:** `Device(config)# arp vrf vrf-name ip-address mac-address ARPA`	Creates a static ARP entry in the specified VRF.

Configuring Per-VRF for TACACS+ Servers

The per-VRF for TACACS+ servers feature enables you to configure per-virtual route forwarding (per-VRF) authentication, authorization, and accounting (AAA) on TACACS+ servers.

You can create the VRF routing table (shown in Steps 3 and 4) and configure the interface (Steps 6, 7, and 8). The actual configuration of per-VRF on a TACACS+ server is done in Steps 10 through 13.

Before you begin

Before configuring per-VRF on a TACACS+ server, you must have configured AAA and a server group.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition vrf-name`	Configures a VRF table and enters VRF configuration mode. You must have a Network Advantage license to configure VRF Definition.
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd route-distinguisher`	Creates routing and forwarding tables for a VRF instance.
Step 5	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode.
Step 6	**interface** *interface-name* **Example:** `Device(config)#  interface GigabitEthernet2/1`	Configures an interface and enters interface configuration mode.
Step 7	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding vrf-name`	Configures a VRF for the interface.
Step 8	**ip address** *ip-address mask* [**secondary**] **Example:** `Device(config-if)# ip address ip-address mask [secondary]`	Sets a primary or secondary IP address for an interface.
Step 9	**exit** **Example:** `Device(config-vrf)# exit`	Exits interface configuration mode.

	Command or Action	Purpose
Step 10	**aaa group server tacacs+** *group-name* **Example:** `Device(config)# aaa group server tacacs+ tacacs1`	Groups different TACACS+ server hosts into distinct lists and distinct methods and enters server-group configuration mode.
Step 11	**server-private** {*ip-address* | *name*} [**nat**] [**single-connection**] [**port** *port-number*] [**timeout** *seconds*] [**key** [**0** | **7**] *string*] **Example:** `Device(config-sg-tacacs+)# server-private 10.1.1.1 port 19 key cisco`	Configures the IP address of the private TACACS+ server for the group server.
Step 12	**vrf forwarding** *vrf-name* **Example:** `Device(config-sg-tacacs+)# vrf forwarding vrf-name`	Configures the VRF reference of a AAA TACACS+ server group.
Step 13	**ip tacacs source-interface** *subinterface-name* **Example:** `Device(config-sg-tacacs+)# ip tacacs source-interface subinterface-name`	Uses the IP address of a specified interface for all outgoing TACACS+ packets.
Step 14	**exit** **Example:** `Device(config-sg-tacacs)# exit`	Exits server-group configuration mode.

Example

The following example lists all the steps to configure per-VRF TACACS+:

```
Device> enable
Device# configure terminal
Device(config)# vrf definition cisco
Device(config-vrf)# rd 100:1
Device(config-vrf)# exit
Device(config)# interface Loopback0
Device(config-if)# vrf forwarding cisco
Device(config-if)# ip address 10.0.0.2 255.0.0.0
Device(config-if)# exit
Device(config-sg-tacacs+)# vrf forwarding cisco
Device(config-sg-tacacs+)# ip tacacs source-interface Loopback0
Device(config-sg-tacacs)# exit
```

Configuring Multicast VRFs

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**ip routing** **Example:** `Device(config)# ip routing`	Enables IP routing.
Step 3	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition vrf-name`	Configures a VRF table and enters VRF configuration mode.
Step 4	**ip multicast-routing vrf** *vrf-name* **Example:** `Device(config-vrf)# ip multicast-routing vrf vrf-name`	(Optional) Enables global multicast routing for VRF table.
Step 5	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd route-distinguisher`	Creates a VRF table by specifying a route distinguisher. Enter either an AS number and an arbitrary number (xxx:y) or an IP address and arbitrary number (A.B.C.D:y).
Step 6	**route-target** {**export** | **import** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf)# route-target {export | import | both} route-target-ext-community`	Creates a list of import, export, or import and export route target communities for the specified VRF. Enter either an AS system number and an arbitrary number (xxx:y) or an IP address and an arbitrary number (A.B.C.D:y). The route-target-ext-community value should be the same as the route-distinguisher value entered in Step 4.
Step 7	**import map** *route-map* **Example:** `Device(config-vrf)# import map route-map`	(Optional) Associates a route map with the VRF.
Step 8	**interface** *interface-id* **Example:** `Device(config)# interface interface-id`	Enters interface configuration mode and specifies the Layer 3 interface to be associated with the VRF. The interface can be a routed port or a SVI.

	Command or Action	Purpose
Step 9	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding vrf-name`	Associates the VRF with the Layer 3 interface.
Step 10	**ip address** *ip-address* **mask** **Example:** `Device(config-if)# ip address ip-address mask`	Configures IP address for the Layer 3 interface.
Step 11	**ip pim sparse-mode** **Example:** `Device(config-if)# ip pim sparse-mode`	Enables PIM on the VRF-associated Layer 3 interface.
Step 12	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 13	**show vrf definition** [**brief** | **detail** | **interfaces**] [*vrf-name*] **Example:** `Device# show vrf definition brief`	Verifies the configuration. Display information about the configured VRFs.
Step 14	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Example

The following example shows how to configure multicast within a VRF table:

```
Device(config)# ip routing
Device(config)# vrf definition multiVrfA
Device(config-vrf)# ip multicast-routing vrf multiVrfA
Device(config-vrf)# interface GigabitEthernet1/1
Device(config-if)# vrf forwarding multiVrfA
Device(config-if)# ip address 172.21.200.203 255.255.255.0
Device(config-if)# ip pim sparse-mode
```

Configuring IPv4 VRFs

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**ip routing** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition vrf-name`	Names the VRF and enters VRF configuration mode.
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd route-distinguisher`	Creates a VRF table by specifying a route distinguisher. Enter either an Autonomous System number number and an arbitrary number (xxx:y) or an IP address and arbitrary number (A.B.C.D:y).
Step 5	**route-target** {**export** | **import** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf)# route-target {export | import | both} route-target-ext-community`	Creates a list of import, export, or import and export route target communities for the specified VRF. Enter either an AS system number and an arbitrary number (xxx:y) or an IP address and an arbitrary number (A.B.C.D:y).
Step 6	**import map** *route-map* **Example:** `Device(config-vrf)# import map route-map`	(Optional) Associates a route map with the VRF.
Step 7	**interface** *interface-id* **Example:** `Device(config-vrf)# interface interface-id`	Enters interface configuration mode and specify the Layer 3 interface to be associated with the VRF. The interface can be a routed port or SVI.
Step 8	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding vrf-name`	Associates the VRF with the Layer 3 interface.
Step 9	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 10	**show vrf definition** [**brief** | **detail** | **interfaces**] [*vrf-name*] **Example:** `Device# show vfr definition [brief | detail | interfaces] [vrf-name]`	Verifies the configuration. Displays information about the configured VRFs.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file. Use the **no vrf definition***vrf-name* global configuration command to delete a VRF and to remove all interfaces from it. Use the **no vrf forwarding** interface configuration command to remove an interface from the VRF.

Configuring VRF-lite for IPv6

This section provides information about configuring VRF-lite for IPv6.

Configuring VRF-Aware Services

IPv6 services can be configured on global interfaces and within the global routing instance. IPv6 services are enhanced to run on multiple routing instances; they are VRF-aware. Any configured VRF in the system can be specified for a VRF-aware service.

VRF-aware services are implemented in platform-independent modules. VRF provides multiple routing instances in Cisco IOS. Each platform has its own limit on the number of VRFs it supports.

VRF-aware services have the following characteristics:

- The user can ping a host in a user-specified VRF.
- Neighbor Discovery entries are learned in separate VRFs. The user can display Neighbor Discovery (ND) entries for specific VRFs.

The following services are VRF-aware:

- Ping
- Unicast Reverse Path Forwarding (uRPF)
- Traceroute
- FTP and TFTP
- Telnet and SSH
- NTP

Configuring the User Interface for PING

Perform the following task to configure a VRF-aware ping:

Procedure

	Command or Action	Purpose
Step 1	**ping vrf** *vrf-name* **ipv6-host** **Example:** `Device# ping vrf vrf-name ipv6-host`	Pings an IPv6 host or address in the specified VRF.

Configuring the User Interface for uRPF

You can configure uRPF on an interface assigned to a VRF. Source lookup is performed in the VRF table

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface interface-id`	Enters interface configuration mode and specifies the Layer 3 interface to configure.
Step 3	**no switchport** **Example:** `Device(config-if)# no switchport`	Removes the interface from Layer 2 configuration mode if it is a physical interface.
Step 4	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding vrf-name`	Configures VRF on the interface.
Step 5	**ipv6 address** *ip-address* **subnet-mask** **Example:** `Device(config-if)# ip address ip-address mask`	Enters the IPv6 address for the interface.
Step 6	**ipv6 verify unicast source reachable-via rx allow-default** **Example:** `Device(config-if)# ipv6 verify unicast source reachable-via rx allow-default`	Enables uRPF on the interface.
Step 7	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	

Configuring the User Interface for Traceroute

Procedure

	Command or Action	Purpose
Step 1	**traceroute vrf** *vrf-name ipv6address* **Example:** `Device# traceroute vrf vrf-name ipv6address`	Specifies the name of a VPN VRF in which to find the destination address.

Configuring the User Interface for Telnet and SSH

Procedure

	Command or Action	Purpose
Step 1	**telnet** *ipv6-address*/**vrf** *vrf-name* **Example:** `Device# telnet ipv6-address/vrf vrf-name`	Connects through Telnet to an IPv6 host or address in the specified VRF.
Step 2	**ssh -l** *username* **-vrf** *vrf-name* **ipv6-host** **Example:** `Device# ssh -l username -vrf  vrf-name ipv6-host`	Connects through SSH to an IPv6 host or address in the specified VRF.

Configuring the User Interface for NTP

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**ntp server vrf** *vrf-name* **ipv6-host** **Example:** `Device(config)# ntp server vrf vrf-name ipv6-host`	Configure the NTP server in the specified VRF.
Step 3	**ntp peer vrf** *vrf-name* **ipv6-host** **Example:**	Configure the NTP peer in the specified VRF.

	Command or Action	Purpose
	`Device(config)# ntp peer vrf vrf-name ipv6-host`	

Configuring IPv6 VRFs

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition vrf-name`	Names the VRF and enters VRF configuration mode.
Step 3	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd route-distinguisher`	(Optional) Creates a VRF table by specifying a route distinguisher. Enter either an Autonomous System number and an arbitrary number (xxx:y) or an IP address and arbitrary number (A.B.C.D:y).
Step 4	**address-family** *ipv4* | *ipv6* **Example:** `Device(config-vrf)# address-family ipv4 | ipv6`	(Optional) IPv4 by default. Configuration MUST for IPv6.
Step 5	**route-target** {**export** | **import** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf)# route-target {export | import | both} route-target-ext-community`	Creates a list of import, export, or import and export route target communities for the specified VRF. Enter either an AS system number and an arbitrary number (xxx:y) or an IP address and an arbitrary number (A.B.C.D:y). **Note** This command is effective only if BGP is running.
Step 6	**exit-address-family** **Example:** `Device(config-vrf)# exit-address-family`	Exits VRF address-family configuration mode and return to VRF configuration mode.
Step 7	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition vrf-name`	Enters VRF configuration mode.

	Command or Action	Purpose
Step 8	**ipv6 multicast multitopology** **Example:** `Device(config-vrf-af)# ipv6 multicast multitopology`	Enables multicast specific RPF topology.
Step 9	**address-family ipv6 multicast** **Example:** `Device(config-vrf)# address-family ipv6 multicast`	Enter multicast IPv6 address-family.
Step 10	**end** **Example:** `Device(config-vrf-af)# end`	Returns to privileged EXEC mode.

Example

This example shows how to configure VRFs:

```
Device(config)#  vrf definition red
Device(config-vrf)# rd 100:1
Device(config-vrf)# address family ipv6
Device(config-vrf-af)# route-target both 200:1
Device(config-vrf)# exit-address-family
Device(config-vrf)# vrf definition red
Device(config-vrf)# ipv6 multicast multitopology
Device(config-vrf)# address-family ipv6 multicast
Device(config-vrf-af)# end
```

Associating Interfaces to the Defined VRFs

Procedure

	Command or Action	Purpose
Step 1	**interface** *interface-id* **Example:** `Device(config-vrf)# interface interface-id`	Enters interface configuration mode and specify the Layer 3 interface to be associated with the VRF. The interface can be a routed port or SVI.
Step 2	**no switchport** **Example:** `Device(config-if)# no switchport`	Removes the interface from configuration mode if it is a physical interface.
Step 3	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding vrf-name`	Associates the VRF with the Layer 3 interface.

	Command or Action	Purpose
Step 4	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enable IPv6 on the interface.
Step 5	**ipv6 address** *ip-address subnet-mask* **Example:** `Device(config-if)# ipv6 address ip-address subnet-mask`	Enters the IPv6 address for the interface.
Step 6	**show ipv6 vrf** [**brief** | **detail** | **interfaces**] [*vrf-name*] **Example:** `Device# show ipv6 vrf [brief | detail | interfaces] [vrf-name]`	Verifies the configuration. Displays information about the configured VRFs.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Example

This example shows how to associate an interface to VRFs:

```
Switch(config-vrf)# interface gigabitethernet1/1
Switch(config-if)# vrf forwarding red
Switch(config-if)# ipv6 enable
Switch(config-if)# ipv6 address 5000::72B/64
```

Populate VRF with Routes via Routing Protocols

This section provides information about populating VRF with routes via routing protocols.

Configuring VRF Static Routes

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix*/*prefix-length* {*ipv6-address* | **interface-type** *interface-number* [*ipv6-address*]}	To configure static routes specific to VRF.

	Command or Action	Purpose
	Example: `Device(config)# ipv6 route [vrf vrf-name] ipv6-prefix/prefix-length {ipv6-address | interface-type interface-number [ipv6-address]}`	

Example

```
Device(config)# ipv6 route vrf v6a 7000::/64 GigabitEthernet 1/1 4000::2
```

Configuring OSPFv3 Router Process

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**router ospfv3** *process-id* **Example:** `Device(config)# router ospfv3 process-id`	Enables OSPFv3 router configuration mode for the IPv6 address family.
Step 3	**area** *area-ID* [**default-cot** | **nssa** | **stub**] **Example:** `Device(config-router)# area area-ID [default-cot | nssa | stub]`	Configures the OSPFv3 area.
Step 4	**router-id** *router-id* **Example:** `Device(config-router)# router-id router-id`	Use a fixed router ID.
Step 5	**address-family ipv6 unicast vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv6 unicast vrf vrf-name`	Enters IPv6 address family configuration mode for OSPFv3 in VRF vrf-name
Step 6	**redistribute source-protocol** [*process-id*] **options** **Example:** `Device(config-router)# redistribute source-protocol [process-id] options`	Redistributes IPv6 routes from one routing domain into another routing domain.
Step 7	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-router)# end`	

Example

This example shows how configure the OSPFv3 router process:

```
Device(config-router)# router ospfv3 1
Device(config-router)# router-id 1.1.1.1
Device(config-router)# address-family ipv6 unicast
Device(config-router-af)# exit-address-family
```

Enabling OSPFv3 on an Interface

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type-number* **Example:** `Device(config)# interface GigabitEthernet2/1`	Specifies an interface type and number, and places the switch in interface configuration mode.
Step 3	**ospfv3** *process-id* **area** *area-ID* **ipv6** [**instance** *instance-id*] **Example:** `Device(config-if)# ospfv3 process-id area area-ID ipv6 [instance instance-id]`	Enables OSPFv3 on an interface with IPv6 AF.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Example

This example show how to enable OSPFv3 on an interface:

```
Device(config)# interface GigabitEthernet2/1
Device(config-if)# no switchport
Device(config-if)# ipv6 address 4000::2/64
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 ospf 1 area 0
Device(config-if)# end
```

Configuring EIGRPv6 Routing Process

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**router eigrp** *virtual-instance-name* **Example:** `Device(config)# router eigrp virtual-instance-name`	Configures the EIGRP routing process and enters router configuration mode.
Step 3	**address-family ipv6 vrf** *vrf-name* **autonomous-system** *autonomous-system-number* **Example:** `Device(config-router)# address-family ipv6 vrf vrf-name autonomous-system autonomous-system-number`	Enables EIGRP IPv6 VRF-Lite and enters address family configuration mode.
Step 4	**topology** {**base** | **topology-name tid number** **Example:** `Device(config-router-af)# topology {base | topology-name tid number`	Configures an EIGRP process to route IP traffic under the specified topology instance and enters address family topology configuration mode.
Step 5	**exit-aftopology** **Example:** `Device(config-router-af-topology)# exit-aftopology`	Exits address family topology configuration mode.
Step 6	**eigrp router-id** *ip-address* **Example:** `Device(config-router)# eigrp router-id ip-address`	Enables the use of a fixed router-id.
Step 7	end **Example:** `Device(config-router)# end`	Exits router configuration mode.

Example

This example shows how to configure an EIGRP routing process:

```
Device(config)# router eigrp test
Device(config-router)# address-family ipv6 unicast vrf b1 autonomous-system 10
Device(config-router-af)# topology base
```

```
Device(config-router-af-topology)# exit-af-topology
Device(config-router)# eigrp router-id 2.3.4.5
Device(config-router)# exit-address-family
```

Additional Information for VRF-lite

This section provides additional information about VRF-lite.

VPN Co-existence Between IPv4 and IPv6

Backward compatibility between the “older” CLI for configuring IPv4 and the “new” CLI for IPv6 exists. This means that a configuration might contain both CLI. The IPv4 CLI retains the ability to have on the same interface, an IP address defined within a VRF as well as an IPv6 address defined in the global routing table.

For example:

```
vrf definition red
 rd 100:1
 address family ipv6
 route-target both 200:1
 exit-address-family
!
vrf definition blue
 rd 200:1
 route-target both 200:1
!
interface GigabitEthernet1/1
 vrf forwarding red
 ip address 50.1.1.2 255.255.255.0
 ipv6 address 4000::72B/64
!
interface GigabitEthernet1/2
 vrf forwarding blue
 ip address 60.1.1.2 255.255.255.0
 ipv6 address 5000::72B/64
```

In this example, all addresses (v4 and v6) defined for GigabitEthernet1/1 refer to VRF red whereas for GigabitEthernet1/2, the IP address refers to VRF blue but the ipv6 address refers to the global IPv6 routing table.

Verifying VRF-lite Configuration

This section provides steps for verifying VRF-lite configuration.

Displaying IPv4 VRF-lite Status

To display information about VRF-lite configuration and status, perform one of the following tasks:

Command	Purpose
`Device# show ip protocols vrf vrf-name`	Displays routing protocol information associated with a VRF.

Command	Purpose
`Device# show ip route vrf vrf-name [connected] [protocol [as-number]] [list] [mobile] [odr] [profile] [static] [summary][supernets-only]`	Displays IP routing table information associated with a VRF.
`Device# show vrf definition [brief | detail | interfaces] [vrf-name]`	Displays information about the defined VRF instances.
`Device# bidir vrf instance-name a.b.c.d | active | bidriectional| count | interface | proxy | pruned | sparse | ssm | static | summary`	Displays information about the defined VRF instances.

This example shows how to display multicast route table information within a VRF instance:

```
Switch# show ip mroute 226.0.0.2
IP Multicast Routing Table
Flags: S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry, E - Extranet,
       X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       G - Received BGP C-Mroute, g - Sent BGP C-Mroute,
       N - Received BGP Shared-Tree Prune, n - BGP C-Mroute suppressed,
       Q - Received BGP S-A Route, q - Sent BGP S-A Route,
       V - RD & Vector, v - Vector, p - PIM Joins on route,
       x - VxLAN group, c - PFP-SA cache created entry
Outgoing interface flags: H - Hardware switched, A - Assert winner, p - PIM Join
Timers: Uptime/Expires
Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 226.0.0.2), 00:01:17/stopped, RP 1.11.1.1, flags: SJCF
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Vlan100, Forward/Sparse, 00:01:17/00:02:36

(5.0.0.11, 226.0.0.2), 00:01:17/00:01:42, flags: FT
  Incoming interface: Vlan5, RPF nbr 0.0.0.0
  Outgoing interface list:
    Vlan100, Forward/Sparse, 00:01:17/00:02:36
```

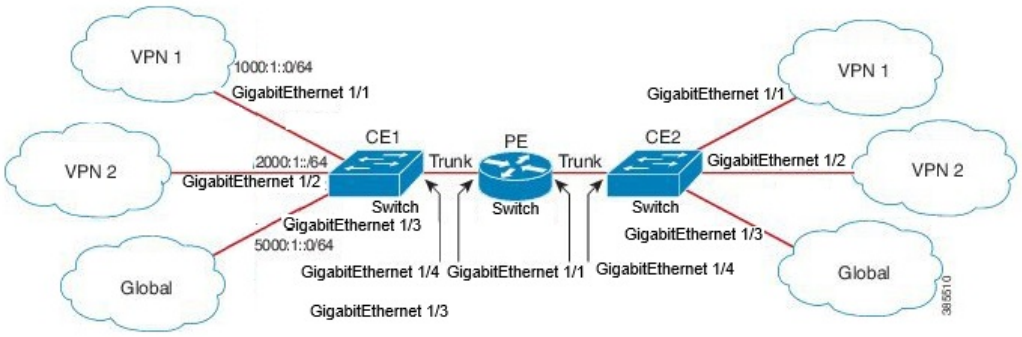
Configuration Examples for VRF-lite

This section provides configuration examples for VRF-lite.

Configuration Example for IPv6 VRF-lite

The following topology illustrates how to use OSPFv3 for CE-PE routing.

Figure 46: VRF-lite Configuration Example



Configuring CE1 Switch

```
ipv6 unicast-routing
vrf definition v1
 rd 100:1
 !
address-family ipv6
 exit-address-family
!

vrf definition v2
 rd 200:1
 !
address-family ipv6
 exit-address-family
!

interface Vlan100
 vrf forwarding v1
 ipv6 address 1000:1::1/64
 ospfv3 100 ipv6 area 0
!

interface Vlan200
 vrf forwarding v2
 ipv6 address 2000:1::1/64
 ospfv3 200 ipv6 area 0
!

interface GigabitEthernet 1/1
switchport access vlan 100
end

interface GigabitEthernet 1/2
switchport access vlan 200
end

interface GigabitEthernet 1/24
 switchport trunk encapsulation dot1q

switchport mode trunk
end

router ospfv3 100
 router-id 10.10.10.10
 !
```

```
 address-family ipv6 unicast vrf v1
  redistribute connected
  area 0 normal
 exit-address-family
!

router ospfv3 200
 router-id 20.20.20.20
 !
 address-family ipv6 unicast vrf v2
  redistribute connected
  area 0 normal
 exit-address-family
!
```

Configuring PE Switch

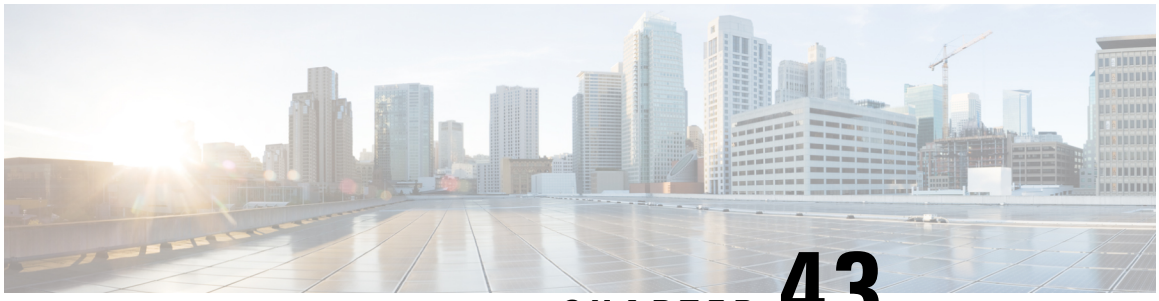
```
ipv6 unicast-routing

vrf definition v1
 rd 100:1
 !
address-family ipv6
 exit-address-family
!

vrf definition v2
 rd 200:1
 !
address-family ipv6
 exit-address-family
!
interface Vlan600
 vrf forwarding v1
 no ipv6 address
 ipv6 address 1000:1::2/64
 ospfv3 100 ipv6 area 0
!

interface Vlan700
 vrf forwarding v2
 no ipv6 address
 ipv6 address 2000:1::2/64
 ospfv3 200 ipv6 area 0
!

interface Vlan800
 vrf forwarding v1
 ipv6 address 3000:1::7/64
 ospfv3 100 ipv6 area 0
!
interface Vlan900
 vrf forwarding v2
 ipv6 address 4000:1::7/64
 ospfv3 200 ipv6 area 0
!

interface GigabitEthernet 1/1
 switchport trunk encapsulation dot1q
 switchport mode trunk
 exit

interface GigabitEthernet 1/2
 switchport trunk encapsulation dot1q
```

```
switchport mode trunk
 exit

router ospfv3 100
 router-id 30.30.30.30
 !
 address-family ipv6 unicast vrf v1
  redistribute connected
  area 0 normal
 exit-address-family
 !
address-family ipv6 unicast vrf v2
  redistribute connected
  area 0 normal
 exit-address-family
 !
```

Configuring CE2 Switch

```
ipv6 unicast-routing

vrf definition v1
 rd 100:1
 !
address-family ipv6
 exit-address-family
!

vrf definition v2
 rd 200:1
 !
address-family ipv6
 exit-address-family
!

interface Vlan100
 vrf forwarding v1

ipv6 address 1000:1::3/64
 ospfv3 100 ipv6 area 0
!

interface Vlan200
 vrf forwarding v2
 ipv6 address 2000:1::3/64
 ospfv3 200 ipv6 area 0
!

interface GigabitEthernet 1/1
switchport access vlan 100
end

interface GigabitEthernet 1/2
switchport access vlan 200
end

interface GigabitEthernet 1/24
 switchport trunk encapsulation dot1q
 switchport mode trunk
 end

router ospfv3 100
 router-id 40.40.40.40
```

```
 !
 address-family ipv6 unicast vrf v1
  redistribute connected
  area 0 normal
 exit-address-family
!

router ospfv3 200
 router-id 50.50.50.50
 !
 address-family ipv6 unicast vrf v2
  redistribute connected

area 0 normal
 exit-address-family
!
```



CHAPTER 43

Configuring Multi-VRF CE

Information About Multi-VRF CE

Virtual Private Networks (VPNs) provide a secure way for customers to share bandwidth over an ISP backbone network. A VPN is a collection of sites sharing a common routing table. A customer site is connected to the service-provider network by one or more interfaces, and the service provider associates each interface with a VPN routing table, called a VPN routing/forwarding (VRF) table.

The switch supports multiple VPN routing/forwarding (multi-VRF) instances in customer edge (CE) devices (multi-VRF CE) when the it is running the . Multi-VRF CE allows a service provider to support two or more VPNs with overlapping IP addresses.

Note The switch does not use Multiprotocol Label Switching (MPLS) to support VPNs.

Understanding Multi-VRF CE

Multi-VRF CE is a feature that allows a service provider to support two or more VPNs, where IP addresses can be overlapped among the VPNs. Multi-VRF CE uses input interfaces to distinguish routes for different VPNs and forms virtual packet-forwarding tables by associating one or more Layer 3 interfaces with each VRF. Interfaces in a VRF can be either physical, such as Ethernet ports, or logical, such as VLAN SVIs, but an interface cannot belong to more than one VRF at any time.

Note Multi-VRF CE interfaces must be Layer 3 interfaces.

Multi-VRF CE includes these devices:

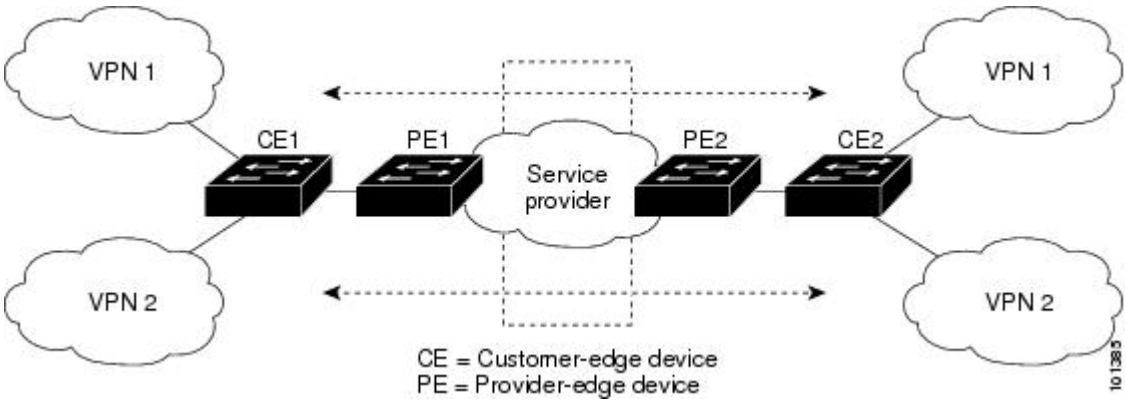
- Customer edge (CE) devices provide customers access to the service-provider network over a data link to one or more provider edge routers. The CE device advertises the site's local routes to the router and learns the remote VPN routes from it. A switch can be a CE.
- Provider routers or core routers are any routers in the service provider network that do not attach to CE devices.

With multi-VRF CE, multiple customers can share one CE, and only one physical link is used between the CE and the PE. The shared CE maintains separate VRF tables for each customer and switches or routes packets for each customer based on its own routing table. Multi-VRF CE extends limited PE functionality to a CE device, giving it the ability to maintain separate VRF tables to extend the privacy and security of a VPN to the branch office.

Network Topology

The figure shows a configuration using switches as multiple virtual CEs. This scenario is suited for customers who have low bandwidth requirements for their VPN service, for example, small companies. In this case, multi-VRF CE support is required in the switches. Because multi-VRF CE is a Layer 3 feature, each interface in a VRF must be a Layer 3 interface.

Figure 47: Switches Acting as Multiple Virtual CEs



When the CE switch receives a command to add a Layer 3 interface to a VRF, it sets up the appropriate mapping between the VLAN ID and the policy label (PL) in multi-VRF-CE-related data structures and adds the VLAN ID and PL to the VLAN database.

When multi-VRF CE is configured, the Layer 3 forwarding table is conceptually partitioned into two sections:

- The multi-VRF CE routing section contains the routes from different VPNs.
- The global routing section contains routes to non-VPN networks, such as the Internet.

VLAN IDs from different VRFs are mapped into different policy labels, which are used to distinguish the VRFs during processing. For each new VPN route learned, the Layer 3 setup function retrieves the policy label by using the VLAN ID of the ingress port and inserts the policy label and new route to the multi-VRF CE routing section. If the packet is received from a routed port, the port internal VLAN ID number is used; if the packet is received from an SVI, the VLAN number is used.

Packet-Forwarding Process

This is the packet-forwarding process in a multi-VRF-CE-enabled network:

- When the switch receives a packet from a VPN, the switch looks up the routing table based on the input policy label number. When a route is found, the switch forwards the packet to the PE.
- When the ingress PE receives a packet from the CE, it performs a VRF lookup. When a route is found, the router adds a corresponding MPLS label to the packet and sends it to the MPLS network.
- When an egress PE receives a packet from the network, it strips the label and uses the label to identify the correct VPN routing table. Then it performs the normal route lookup. When a route is found, it forwards the packet to the correct adjacency.
- When a CE receives a packet from an egress PE, it uses the input policy label to look up the correct VPN routing table. If a route is found, it forwards the packet within the VPN.

Network Components

To configure VRF, you create a VRF table and specify the Layer 3 interface associated with the VRF. Then configure the routing protocols in the VPN and between the CE and the PE. The multi-VRF CE network has three major components:

- VPN route target communities—lists of all other members of a VPN community. You need to configure VPN route targets for each VPN community member.
- VPN forwarding—transports all traffic between all VPN community members across a VPN service-provider network.

VRF-Aware Services

IP services can be configured on global interfaces, and these services run within the global routing instance. IP services are enhanced to run on multiple routing instances; they are VRF-aware. Any configured VRF in the system can be specified for a VRF-aware service.

VRF-Aware services are implemented in platform-independent modules. VRF means multiple routing instances in Cisco IOS. Each platform has its own limit on the number of VRFs it supports.

VRF-aware services have the following characteristics:

- The user can ping a host in a user-specified VRF.
- ARP entries are learned in separate VRFs. The user can display Address Resolution Protocol (ARP) entries for specific VRFs.

Multi-VRF CE Configuration Guidelines

This section provides guidelines for configuring multi-VRF CE:

- A switch with multi-VRF CE is shared by multiple customers, and each customer has its own routing table.
- Because customers use different VRF tables, the same IP addresses can be reused. Overlapped IP addresses are allowed in different VPNs.
- Multi-VRF CE lets multiple customers share the same physical link between the PE and the CE. Trunk ports with multiple VLANs separate packets among customers. Each customer has its own VLAN.

- Multi-VRF CE does not support all MPLS-VRF functionality. It does not support label exchange, LDP adjacency, or labeled packets.
- For the PE router, there is no difference between using multi-VRF CE or using multiple CEs. In Figure 41-6, multiple virtual Layer 3 interfaces are connected to the multi-VRF CE device.
- The switch supports configuring VRF by using physical ports, VLAN SVIs, or a combination of both. The SVIs can be connected through an access port or a trunk port.
- A customer can use multiple VLANs as long as they do not overlap with those of other customers. A customer's VLANs are mapped to a specific routing table ID that is used to identify the appropriate routing tables stored on the switch.
- Multi-VRF CE does not affect the packet switching rate.
- VPN multicast is not supported.
- You can enable VRF on a private VLAN, and the reverse.
- You cannot enable VRF when policy-based routing (PBR) is enabled on an interface, and the reverse.
- You cannot enable VRF when Web Cache Communication Protocol (WCCP) is enabled on an interface, and the reverse.

How to Configure Multi-VRF CE

The following sections provide configurational information about Multi-VRF CE.

Default Multi-VRF CE Configuration

Table 53: Default VRF Configuration

Feature	Default Setting
VRF	Disabled. No VRFs are defined.
Maps	No import maps, export maps, or route maps are defined.
VRF maximum routes	Gigabit Ethernet switches: 12000.
Forwarding table	The default for an interface is the global routing table.

Configuring VRFs

Perform the following steps:

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device>enable`	Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**ip routing** **Example:** `Device(config)#ip routing`	Enables IP routing.
Step 4	**ip vrf** *vrf-name* **Example:** `Device(config)#ip vrf vpn1`	Names the VRF, and enter VRF configuration mode.
Step 5	**rd** *route-distinguisher* **Example:** `Device(config-vrf)#rd 100:2`	Creates a VRF table by specifying a route distinguisher. Enter either an AS number and an arbitrary number (xxx:y) or an IP address and arbitrary number (A.B.C.D:y)
Step 6	**route-target** {**export** | **import** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf)#route-target both 100:2`	Creates a list of import, export, or import and export route target communities for the specified VRF. Enter either an AS system number and an arbitrary number (xxx:y) or an IP address and an arbitrary number (A.B.C.D:y). The *route-target-ext-community* should be the same as the *route-distinguisher* entered in Step 4.
Step 7	**import map** *route-map* **Example:** `Device(config-vrf)#import map importmap1`	(Optional) Associates a route map with the VRF.
Step 8	**interface** *interface-id* **Example:** `Device(config-vrf)#interface gigabitethernet 1/1`	Specifies the Layer 3 interface to be associated with the VRF, and enter interface configuration mode. The interface can be a routed port or SVI.
Step 9	**ip vrf forwarding** *vrf-name* **Example:**	Associates the VRF with the Layer 3 interface. **Note**

	Command or Action	Purpose
	`Device(config-if)#ip vrf forwarding vpn1`	When **ip vrf forwarding** is enabled in the Management Interface, the access point does not join.
Step 10	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 11	**show ip vrf** [**brief** | **detail** | **interfaces**] [*vrf-name*] **Example:** `Device#show ip vrf interfaces vpn1`	Verifies the configuration. Displays information about the configured VRFs.
Step 12	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Multicast VRFs

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**ip routing** **Example:** `Device(config)#ip routing`	Enables IP routing mode.

	Command or Action	Purpose
Step 4	**ip vrf** *vrf-name* **Example:** `Device(config)#ip vrf vpn1`	Names the VRF, and enter VRF configuration mode.
Step 5	**rd** *route-distinguisher* **Example:** `Device(config-vrf)#rd 100:2`	Creates a VRF table by specifying a route distinguisher. Enter either an AS number and an arbitrary number (xxx:y) or an IP address and an arbitrary number (A.B.C.D:y)
Step 6	**route-target** {**export** | **import** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf)#route-target import 100:2`	Creates a list of import, export, or import and export route target communities for the specified VRF. Enter either an AS system number and an arbitrary number (xxx:y) or an IP address and an arbitrary number (A.B.C.D:y). The *route-target-ext-community* should be the same as the *route-distinguisher* entered in Step 4.
Step 7	**import map** *route-map* **Example:** `Device(config-vrf)#import map importmap1`	(Optional) Associates a route map with the VRF.
Step 8	**ip multicast-routing vrf** *vrf-name* **distributed** **Example:** `Device(config-vrf)#ip multicast-routing vrf vpn1 distributed`	(Optional) Enables global multicast routing for VRF table.
Step 9	**interface** *interface-id* **Example:** `Device(config-vrf)#interface gigabitethernet 1/2`	Specifies the Layer 3 interface to be associated with the VRF, and enter interface configuration mode. The interface can be a routed port or an SVI.
Step 10	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)#ip vrf forwarding vpn1`	Associates the VRF with the Layer 3 interface.
Step 11	**ip address** *ip-address* **mask** **Example:** `Device(config-if)#ip address 10.1.5.1 255.255.255.0`	Configures IP address for the Layer 3 interface.

	Command or Action	Purpose
Step 12	**ip pim sparse-dense mode** **Example:** `Device(config-if)#ip pim sparse-dense mode`	Enables PIM on the VRF-associated Layer 3 interface.
Step 13	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 14	**show ip vrf** [**brief** | **detail** | **interfaces**] [*vrf-name*] **Example:** `Device#show ip vrf detail vpn1`	Verifies the configuration. Displays information about the configured VRFs.
Step 15	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a VPN Routing Session

Routing within the VPN can be configured with any supported routing protocol (RIP, OSPF, EIGRP, or) or with static routing. The configuration shown here is for OSPF, but the process is the same for other protocols.

Note To configure an EIGRP routing process to run within a VRF instance, you must configure an autonomous-system number by entering the **autonomous-system** *autonomous-system-number* address-family configuration mode command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**router ospf** *process-id* **vrf** *vrf-name* **Example:** `Device(config)#router ospf 1 vrf vpn1`	Enables OSPF routing, specifies a VPN forwarding table, and enter router configuration mode.
Step 4	**log-adjacency-changes** **Example:** `Device(config-router)#log-adjacency-changes`	(Optional) Logs changes in the adjacency state. This is the default state.
Step 5	**redistribute isis subnets** **Example:** `Device(config-router)#redistribute isis 10 subnets`	Sets the switch to redistribute information from the ISIS network to the OSPF network.
Step 6	**network** *network-number* **area** *area-id* **Example:** `Device(config-router)#network 1 area 2`	Defines a network address and mask on which OSPF runs and the area ID for that network address.
Step 7	**end** **Example:** `Device(config-router)#end`	Returns to privileged EXEC mode.
Step 8	**show ip ospf** *process-id* **Example:** `Device#show ip ospf 1`	Verifies the configuration of the OSPF network.
Step 9	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring VRF-Aware Services

These services are VRF-Aware:

- ARP

- Ping
- Simple Network Management Protocol (SNMP)
- Unicast Reverse Path Forwarding (uRPF)
- Syslog
- Traceroute
- FTP and TFTP

Configuring VRF-Aware Services for SNMP

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**snmp-server trap authentication vrf** **Example:** `Device(config)#snmp-server trap authentication vrf`	Enables SNMP traps for packets on a VRF.
Step 4	**snmp-server engineID remote** *host* **vrf** *vpn-instance engine-id string* **Example:** `Device(config)#snmp-server engineID remote 172.16.20.3 vrf vpn1 80000009030000B064EFE100`	Configures a name for the remote SNMP engine on a switch.
Step 5	**snmp-server host** *host* **vrf** *vpn-instance* **traps** *community* **Example:** `Device(config)#snmp-server host 172.16.20.3 vrf vpn1 traps comaccess`	Specifies the recipient of an SNMP trap operation and specifies the VRF table to be used for sending SNMP traps.

	Command or Action	Purpose
Step 6	**snmp-server host** *host* **vrf** *vpn-instance* **informs** *community* **Example:** `Device(config)#snmp-server host 172.16.20.3 vrf vpn1 informs comaccess`	Specifies the recipient of an SNMP inform operation and specifies the VRF table to be used for sending SNMP informs.
Step 7	**snmp-server user** *user group* **remote** *host* **vrf** *vpn-instance security model* **Example:** `Device(config)#snmp-server user abcd remote 172.16.20.3 vrf vpn1 priv v2c 3des secure3des`	Adds a user to an SNMP group for a remote host on a VRF for SNMP access.
Step 8	**end** **Example:** `Device(config-if)#end`	Returns to privileged EXEC mode.

Configuring VRF-Aware Services for NTP

Configuring VRF-aware services for NTP comprises configuring the NTP servers and the NTP client interfaces connected to the NTP servers.

Before you begin

Ensure connectivity between the NTP client and servers. Configure a valid IP address and subnet on the client interfaces that are connected to the NTP servers.

Configuring VRF-Aware Services for NTP on NTP Client

Perform the following steps on the client interface that is connected to the NTP server.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device#configure terminal`	
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Specifies the Layer 3 interface to be associated with the VRF, and enters the interface configuration mode.
Step 4	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)#vrf forwarding A`	Associates the VRF with the Layer 3 interface.
Step 5	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)#ip address 1.1.1.1 255.255.255.0`	Enter the IP address for the interface.
Step 6	**no shutdown** **Example:** `Device(config-if)#no shutdown`	Enables the interface.
Step 7	**exit** **Example:** `Device(config-if)exit`	Exits the interface configuration mode.
Step 8	**ntp authentication-key** *number* **md5** *md5-number* **Example:** `Device(config)#ntp authentication-key 1 md5 cisco123`	Defines the authentication keys. The device does not synchronize to a time source unless the source has one of these authentication keys and the key number is specified by the **ntp trusted-key number** command. **Note** The authentication key *number* and the MD5 *passowrd* must be the same on both the client and server.
Step 9	**ntp authenticate** **Example:** `Device(config)#ntp authenticate`	Enables the NTP authentication feature. NTP authentication is disabled by default.
Step 10	**ntp trusted-key** *key-number* **Example:** `Device(config)#ntp trusted-key 1`	Specifies one or more keys that an NTP server must provide in its NTP packets in order for the NTP client to synchronize to it. The range for trusted keys is from 1 to 65535. This command provides protection against accidentally synchronizing the NTP client to an NTP server that is not trusted.

	Command or Action	Purpose
Step 11	**ntp server vrf** *vrf-name* **Example:** `Device(config)#ntp server vrf A 1.1.1.2 key 1`	Configures NTP server in the specified VRF.

Configuring VRF-Aware Services for NTP on the NTP Server

Perform the following steps on the NTP server.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**ntp authentication-key** *number* **md5** *passowrd* **Example:** `Device(config)#ntp authentication-key 1 md5 cisco123`	Defines the authentication keys. The device does not synchronize to a time source unless the source has one of these authentication keys and the key number is specified by the **ntp trusted-key number** command. **Note** The authentication key *number* and the MD5 *passowrd* must be the same on both the client and server.
Step 4	**ntp authenticate** **Example:** `Device(config)#ntp authenticate`	Enables the NTP authentication feature. NTP authentication is disabled by default.
Step 5	**ntp trusted-key** *key-number* **Example:** `Device(config)#ntp trusted-key 1`	Specifies one or more keys that an NTP server must provide in its NTP packets in order for the NTP client to synchronize to it. The range for trusted keys is from 1 to 65535. This command provides protection against accidentally synchronizing the NTP client to an NTP server that is not trusted.

	Command or Action	Purpose
Step 6	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/3`	Specifies the Layer 3 interface to be associated with the VRF, and enters the interface configuration mode.
Step 7	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)#vrf forwarding A`	Associates the VRF with the Layer 3 interface.
Step 8	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)#ip address 1.1.1.2 255.255.255.0`	Enter the IP address for the interface.
Step 9	**exit** **Example:** `Device(config-if)exit`	Exits the interface configuration mode.

Configuring VRF-Aware Services for uRPF

uRPF can be configured on an interface assigned to a VRF, and source lookup is done in the VRF table.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 4	**no switchport** **Example:** `Device(config-if)#no switchport`	Removes the interface from Layer 2 configuration mode if it is a physical interface.

	Command or Action	Purpose
Step 5	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)#ip vrf forwarding vpn2`	Configures VRF on the interface.
Step 6	**ip address** *ip-address* **Example:** `Device(config-if)#ip address 10.1.5.1`	Enters the IP address for the interface.
Step 7	**ip verify unicast reverse-path** **Example:** `Device(config-if)#ip verify unicast reverse-path`	Enables uRPF on the interface.
Step 8	**end** **Example:** `Device(config-if)#end`	Returns to privileged EXEC mode.

Configuring VRF-Aware RADIUS

To configure VRF-Aware RADIUS, you must first enable AAA on a RADIUS server. The switch supports the **ip vrf forwarding** *vrf-name* server-group configuration and the **ip radius source-interface** global configuration commands, as described in the *Per VRF AAA Feature Guide*.

Configuring VRF-Aware Services for Syslog

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**logging on** **Example:** `Device(config)#logging on`	Enables or temporarily disables logging of storage router event message.
Step 4	**logging host** *ip-address* **vrf** *vrf-name* **Example:** `Device(config)#logging host 10.10.1.0 vrf vpn1`	Specifies the host address of the syslog server where logging messages are to be sent.
Step 5	**logging buffered** *logging buffered size* **debugging** **Example:** `Device(config)#logging buffered critical 6000 debugging`	Logs messages to an internal buffer.
Step 6	**logging trap debugging** **Example:** `Device(config)#logging trap debugging`	Limits the logging messages sent to the syslog server.
Step 7	**logging facility** *facility* **Example:** `Device(config)#logging facility user`	Sends system logging messages to a logging facility.
Step 8	**end** **Example:** `Device(config-if)#end`	Returns to privileged EXEC mode.

Configuring VRF-Aware Services for Traceroute

Procedure

	Command or Action	Purpose
Step 1	**traceroute vrf** *vrf-name ipaddress* **Example:** `Device(config)#traceroute vrf vpn2 10.10.1.1`	Specifies the name of a VPN VRF in which to find the destination address.

Configuring VRF-Aware Services for FTP and TFTP

So that FTP and TFTP are VRF-aware, you must configure some FTP/TFTP CLIs. For example, if you want to use a VRF table that is attached to an interface, say E1/0, you need to configure the **ip tftp source-interface E1/0** or the **ip ftp source-interface E1/0** command to inform TFTP or FTP server to use a specific routing table. In this example, the VRF table is used to look up the destination IP address. These changes are backward-compatible and do not affect existing behavior. That is, you can use the source-interface CLI to send packets out a particular interface even if no VRF is configured on that interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**ip ftp source-interface** *interface-type interface-number* **Example:** `Device(config)#ip ftp source-interface gigabitethernet 1/2`	Specifies the source IP address for FTP connections.
Step 4	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.
Step 5	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 6	**ip tftp source-interface** *interface-type interface-number* **Example:** `Device(config)#ip tftp source-interface gigabitethernet 1/2`	Specifies the source IP address for TFTP connections.

	Command or Action	Purpose
Step 7	**end** **Example:** `Device(config)#end`	Returns to privileged EXEC mode.

Monitoring VRF-Aware Services for ARP

Procedure

	Command or Action	Purpose
Step 1	**show ip arp vrf** *vrf-name* **Example:** `Device#show ip arp vrf vpn1`	Displays the ARP table in the specified VRF.

Monitoring VRF-Aware Services for Ping

Procedure

	Command or Action	Purpose
Step 1	**ping vrf***vrf-name***ip-host** **Example:** `Device#ping vrf vpn1 ip-host`	Displays the ARP table in the specified VRF.

Monitoring Multi-VRF CE

This section provides information on commands for monitoring multi-VRF CE:

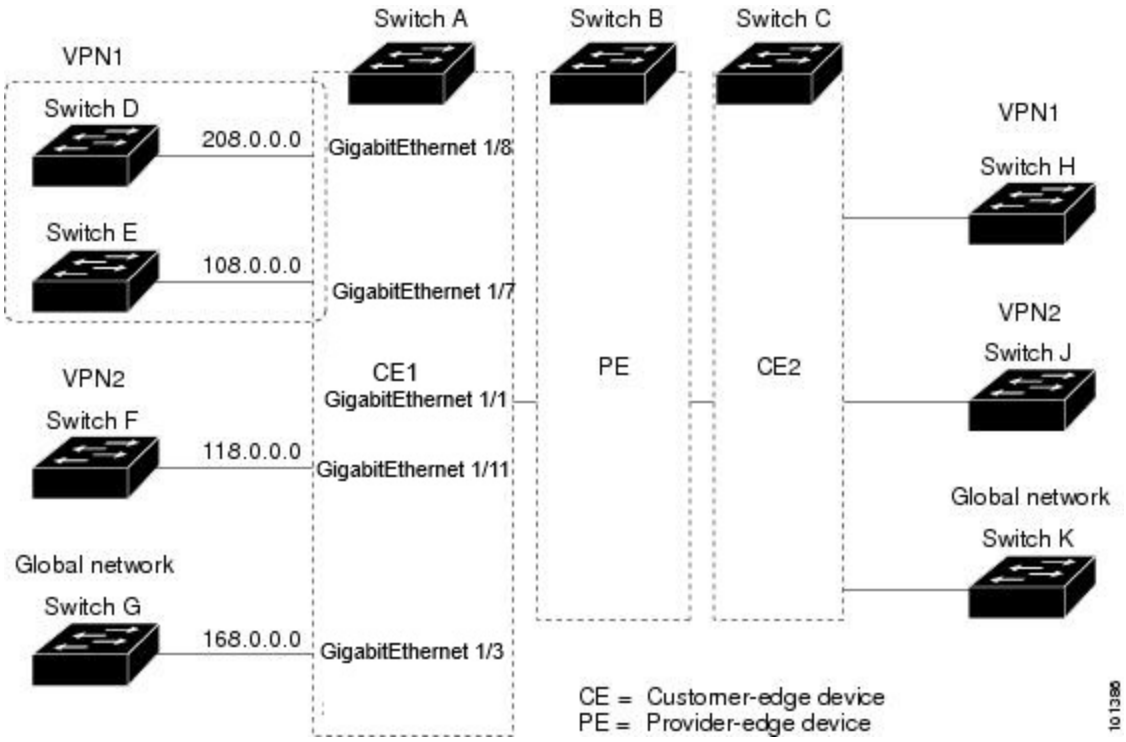
Table 54: Commands for Displaying Multi-VRF CE Information

Command	Purpose
show ip protocols vrf *vrf-name*	Displays routing protocol information associated with a VRF.
show ip route vrf *vrf-name* [**connected**] [*protocol* [*as-number*]] [**list**] [**mobile**] [**odr**] [**profile**] [**static**] [**summary**] [**supernets-only**]	Displays IP routing table information associated with a VRF.
show ip vrf [**brief** | **detail** | **interfaces**] [*vrf-name*]	Displays information about the defined VRF instances.

Configuration Example: Multi-VRF CE

OSPF is the protocol used in VPN1, VPN2, and the global network. The examples following the illustration show how to configure a switch as CE Switch A, and the VRF configuration for customer switches D and F. Commands for configuring CE Switch C and the other customer switches are not included but would be similar.

Figure 48: Establishing a Multi-VRF CE Configuration Example



On Switch A, enable routing and configure VRF.

```
Device#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)#ip routing
Device(config)#ip vrf v11
Device(config-vrf)#rd 800:1
Device(config-vrf)#route-target export 800:1
Device(config-vrf)#route-target import 800:1
Device(config-vrf)#exit
Device(config)#ip vrf v12
Device(config-vrf)#rd 800:2
Device(config-vrf)#route-target export 800:2
Device(config-vrf)#route-target import 800:2
Device(config-vrf)#exit
```

Configure the loopback and physical interfaces on Switch A. Gigabit Ethernet port 1 is a trunk connection to the PE. Gigabit Ethernet ports 8 and 11 connect to VPNs:

```
Device(config)#interface loopback1
Device(config-if)#ip vrf forwarding v11
```

```
Device(config-if)#ip address 8.8.1.8 255.255.255.0
Device(config-if)#exit

Device(config)#interface loopback2
Device(config-if)#ip vrf forwarding v12
Device(config-if)#ip address 8.8.2.8 255.255.255.0
Device(config-if)#exit

Device(config)#interface gigabitethernet1/5
Device(config-if)#switchport trunk encapsulation dot1q
Device(config-if)#switchport mode trunk
Device(config-if)#no ip address
Device(config-if)#exit
Device(config)#interface gigabitethernet1/8
Device(config-if)#switchport access vlan 208
Device(config-if)#no ip address
Device(config-if)#exit
Device(config)#interface gigabitethernet1/11
Device(config-if)#switchport trunk encapsulation dot1q
Device(config-if)#switchport mode trunk
Device(config-if)#no ip address
Device(config-if)#exit
```

Configure the VLANs used on Switch A. VLAN 10 is used by VRF 11 between the CE and the PE. VLAN 20 is used by VRF 12 between the CE and the PE. VLANs 118 and 208 are used for the VPNs that include Switch F and Switch D, respectively:

```
Device(config)#interface vlan10
Device(config-if)#ip vrf forwarding v11
Device(config-if)#ip address 38.0.0.8 255.255.255.0
Device(config-if)#exit
Device(config)#interface vlan20
Device(config-if)#ip vrf forwarding v12
Device(config-if)#ip address 83.0.0.8 255.255.255.0
Device(config-if)#exit
Device(config)#interface vlan118
Device(config-if)#ip vrf forwarding v12
Device(config-if)#ip address 118.0.0.8 255.255.255.0
Device(config-if)#exit
Device(config)#interface vlan208
Device(config-if)#ip vrf forwarding v11
Device(config-if)#ip address 208.0.0.8 255.255.255.0
Device(config-if)#exit
```

Configure OSPF routing in VPN1 and VPN2.

```
Device(config)#router ospf 1 vrf vl1
Device(config-router)#redistribute isis subnets
Device(config-router)#network 208.0.0.0 0.0.0.255 area 0
Device(config-router)#exit
Device(config)#router ospf 2 vrf vl2
Device(config-router)#redistribute isis subnets
Device(config-router)#network 118.0.0.0 0.0.0.255 area 0
Device(config-router)#exit
```

Switch D belongs to VPN 1. Configure the connection to Switch A by using these commands.

```
Device#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)#ip routing
Device(config)#interface gigabitethernet1/2
Device(config-if)#no switchport
```

```
Device(config-if)#ip address 208.0.0.20 255.255.255.0
Device(config-if)#exit

Device(config)#router ospf 101
Device(config-router)#network 208.0.0.0 0.0.0.255 area 0
Device(config-router)#end
```

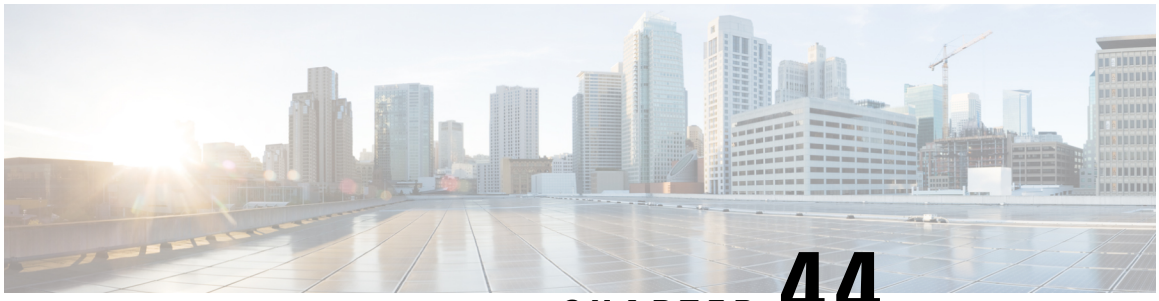
Switch F belongs to VPN 2. Configure the connection to Switch A by using these commands.

```
Device#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)#ip routing
Device(config)#interface gigabitethernet1/1
Device(config-if)#switchport trunk encapsulation dot1q
Device(config-if)#switchport mode trunk
Device(config-if)#no ip address
Device(config-if)#exit

Device(config)#interface vlan118
Device(config-if)#ip address 118.0.0.11 255.255.255.0
Device(config-if)#exit

Device(config)#router ospf 101
Device(config-router)#network 118.0.0.0 0.0.0.255 area 0
Device(config-router)#end
```

When used on switch B (the PE router), these commands configure only the connections to the CE device, Switch A.

```
Device#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)#ip vrf v1
Device(config-vrf)#rd 100:1
Device(config-vrf)#route-target export 100:1
Device(config-vrf)#route-target import 100:1
Device(config-vrf)#exit

Device(config)#ip vrf v2
Device(config-vrf)#rd 100:2
Device(config-vrf)#route-target export 100:2
Device(config-vrf)#route-target import 100:2
Device(config-vrf)#exit
Device(config)#ip cef
Device(config)#interface Loopback1
Device(config-if)#ip vrf forwarding v1
Device(config-if)#ip address 3.3.1.3 255.255.255.0
Device(config-if)#exit

Device(config)#interface Loopback2
Device(config-if)#ip vrf forwarding v2
Device(config-if)#ip address 3.3.2.3 255.255.255.0
Device(config-if)#exit

Device(config)#interface gigabitethernet1/1
Device(config-if)#encapsulation dot1q 10
Device(config-if)#ip vrf forwarding v1
Device(config-if)#ip address 38.0.0.3 255.255.255.0
Device(config-if)#exit

Device(config)#interface gigabitethernet1/1
Device(config-if)#encapsulation dot1q 20
Device(config-if)#ip vrf forwarding v2
Device(config-if)#ip address 83.0.0.3 255.255.255.0
```

```
Device(config-if)#exit

Device(config)#router bgp 100
Device(config-router)#address-family ipv4 vrf v2
Device(config-router-af)#neighbor 83.0.0.8 remote-as 800
Device(config-router-af)#neighbor 83.0.0.8 activate
Device(config-router-af)#network 3.3.2.0 mask 255.255.255.0
Device(config-router-af)#exit
Device(config-router)#address-family ipv4 vrf vl
Device(config-router-af)#neighbor 38.0.0.8 remote-as 800
Device(config-router-af)#neighbor 38.0.0.8 activate
Device(config-router-af)#network 3.3.1.0 mask 255.255.255.0
Device(config-router-af)#end
```



CHAPTER 44

Configuring Unicast Reverse Path Forwarding

Prerequisites for Unicast Reverse Path Forwarding

- Unicast Reverse Path Forwarding (RPF) requires Cisco Express Forwarding to function properly on a device.
- Prior to configuring Unicast RPF, you must configure the following access control lists (ACLs):
  - Configure standard or extended ACL to mitigate the transmission of invalid IP addresses (by performing egress filtering). Configuring standard or extended ACLs permit only valid source addresses to leave your network and enter the Internet.
  - Configure standard or extended ACL entries to drop (deny) packets that have invalid source IP addresses (by performing ingress filtering). Invalid source IP addresses include the following types:
    - Broadcast addresses (including multicast addresses)
    - Loopback addresses
    - Private addresses (RFC 1918, *Address Allocation for Private Internets*)
    - Reserved addresses
    - Source addresses that fall outside the range of valid addresses that are associated with the protected network

Restrictions for Unicast Reverse Path Forwarding

The following basic restrictions apply to multihomed clients:

- Clients should not be multihomed on the same device because multihoming defeats the purpose of creating a redundant service for a client.
- Ensure that packets that flow up the link (out to the Internet) match the route advertised out of the link. Otherwise, Unicast RPF filters these packets as malformed packets.

Information About Unicast Reverse Path Forwarding

The Unicast Reverse Path Forwarding feature helps to mitigate problems that are caused by the introduction of malformed or forged (spoofed) IP source addresses into a network by discarding IP packets that lack a verifiable IP source address. For example, a number of common types of denial-of-service (DoS) attacks, including Smurf and Tribal Flood Network (TFN), can take advantage of forged or rapidly changing source IP addresses to allow attackers to thwart efforts to locate or filter the attacks. For Internet service providers (ISPs) that provide public access, Unicast RPF deflects such attacks by forwarding only packets that have source addresses that are valid and consistent with the IP routing table. This action protects the network of the ISP, its customer, and the rest of the Internet.

Note Enabling IPv4 unicast RPF also enables IPv6 unicast RPF. This is applicable only for the .

Unicast RPF Operation

When Unicast RPF is enabled on an interface of a device, the device examines all packets received as input on that interface to ensure that the source address and source interface information appears in the routing table and matches the interface on which packets are received. This ability to “look backwards” is available only when Cisco Express Forwarding is enabled on a device because the lookup relies on the presence of a Forwarding Information Base (FIB). Cisco Express Forwarding generates a FIB as part of its operation.

Note Unicast RPF is an input function and is applied only on the input interface of a device at the upstream end of a connection.

Unicast RPF does a reverse lookup in the Cisco Express Forwarding table to check if any packet received at the interface of a device arrives on the best return path (or return route) to the source of the packet. If the packet was received from one of the best reverse path routes, the packet is forwarded as normal. No reverse path route on the interface from which the packet was received can mean that the source address was modified. If Unicast RPF cannot find a reverse path for the packet, the packet is dropped or forwarded, depending on whether an access control list (ACL) is specified by using the **ip verify unicast reverse-path** command in interface configuration mode.

Note With Unicast RPF, all equal-cost “best” return paths are considered valid. Unicast RPF supports multiple return paths, provided that each path is equal to the others in terms of the routing cost (such as number of hops, weights, and so on) and the route is available in the FIB. Unicast RPF also functions where Enhanced Interior Gateway Routing Protocol (EIGRP) variants are used.

Before forwarding a packet that is received at the interface on which Unicast RPF and ACLs have been configured, Unicast RPF does the following checks:

1. If input ACLs are configured on the inbound interface.
2. If the packet has arrived on the best return path to the source by doing a reverse lookup in the FIB table.
3. Does a lookup of the Cisco Express Forwarding table for packet forwarding.
4. Checks output ACLs on the outbound interface.
5. Forwards the packet.

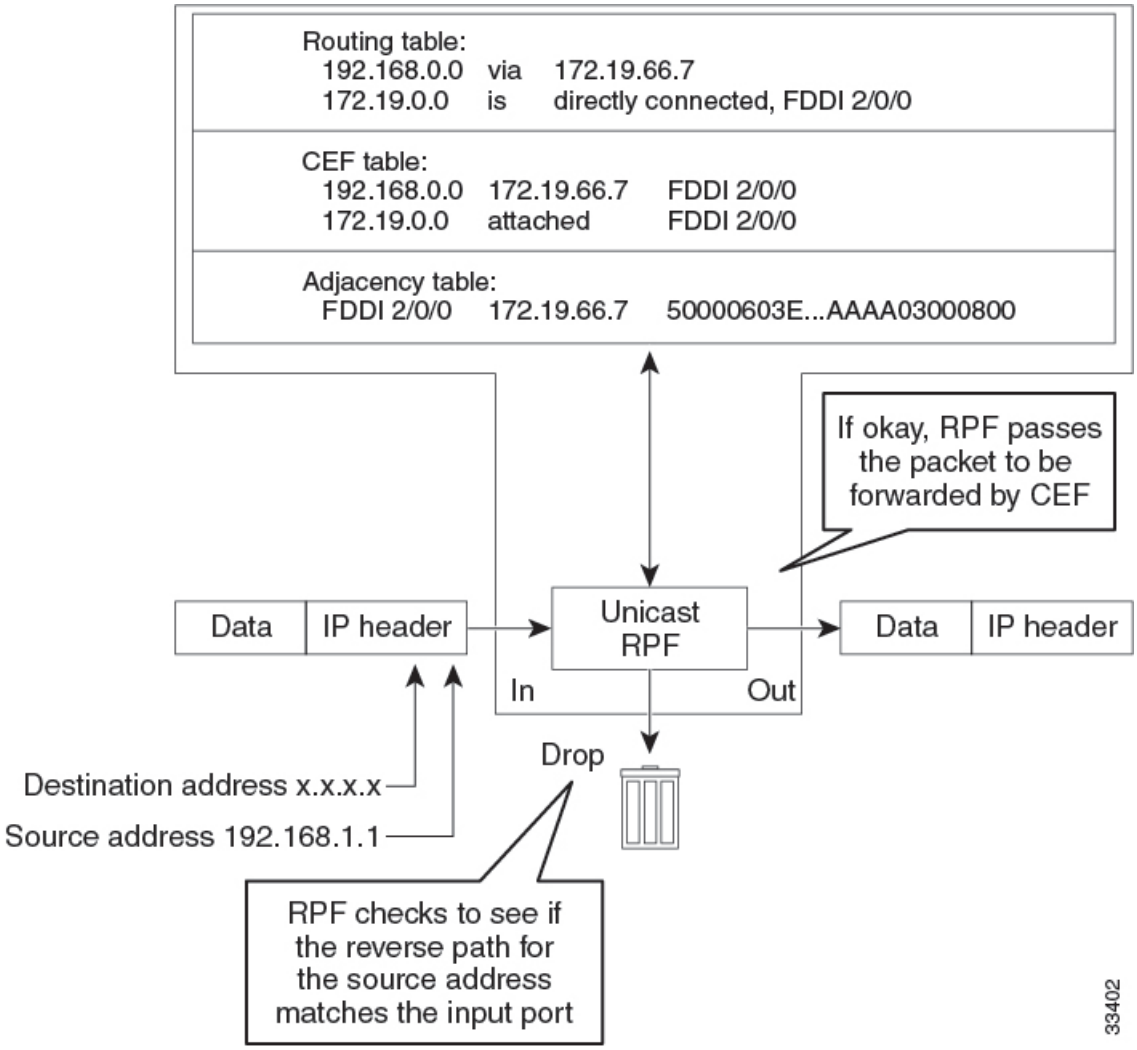
Per-Interface Statistics

Each time a packet is dropped or forwarded at an interface, that information is counted two ways: globally on the device and at each interface where you have applied Unicast RPF. Global statistics on dropped packets provide information about potential attacks on the network; however, these global statistics do not help to specify which interface is the source of the attack.

Per-interface statistics allow network administrators to track two types of information about malformed packets: Unicast RPF drops and Unicast RPF suppressed drops. Statistics on the number of packets that Unicast RPF drops help to identify the interface that is the entry point of the attack. The Unicast RPF drop count tracks the number of drops at the interface. The Unicast RPF suppressed drop count tracks the number of packets that failed the Unicast RPF check but were forwarded because of the permit permission set up in the ACL. Using the drop count and suppressed drop count statistics, a network administrator can takes steps to isolate the attack at a specific interface.

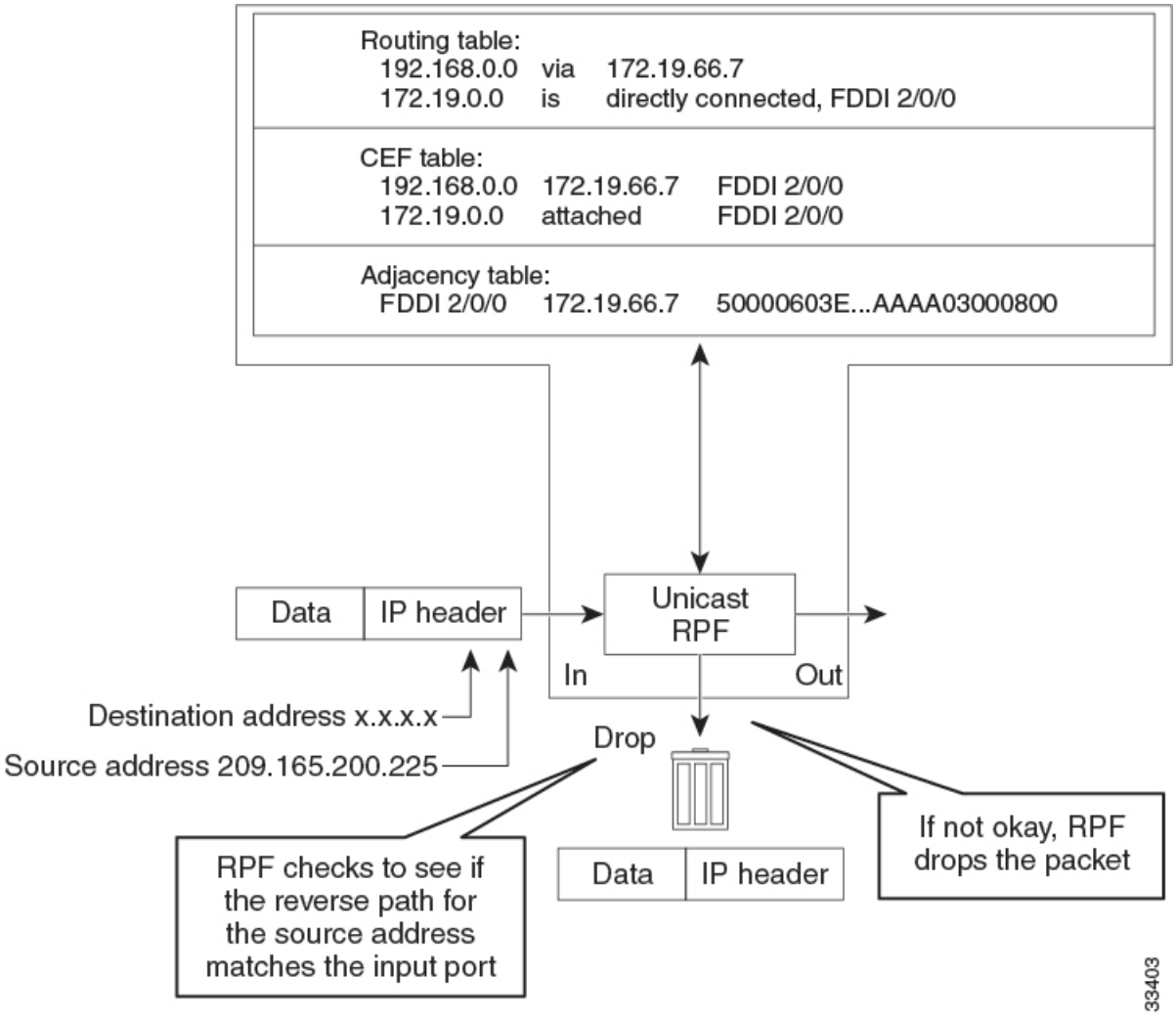
The figure below illustrates how Unicast RPF and CEF work together to validate IP source addresses by verifying packet return paths. In this example, a customer has sent a packet having a source address of 192.168.1.1 from interface FDDI 2/0/0. Unicast RPF checks the FIB to see if 192.168.1.1 has a path to FDDI 2/0/0. If there is a matching path, the packet is forwarded. If there is no matching path, the packet is dropped.

Figure 49: Unicast RPF Validating IP Source Addresses



The figure below illustrates how Unicast RPF drops packets that fail validation. In this example, a customer has sent a packet having a source address of 209.165.200.225, which is received at interface FDDI 2/0/0. Unicast RPF checks the FIB to see if 209.165.200.225 has a return path to FDDI 2/0/0. If there is a matching path, the packet is forwarded. In this case, there is no reverse entry in the routing table that routes the customer packet back to source address 209.165.200.225 on interface FDDI 2/0/0, and so the packet is dropped.

Figure 50: Unicast RPF Dropping Packets That Fail Verification



Implementation of Unicast Reverse Path Forwarding Notification

Unicast RPF is a security feature that verifies the validity of the source IP of an incoming packet. When a packet arrives at an interface and its source IP is unknown in the routing table or is a known bad source address, Unicast RPF drops the packet. IP verification of the source is done to prevent the DoS attacks by detecting problems with the incoming packets on an interface. However, deploying Unicast RPF without some automated monitoring capability is a challenge.

The CISCO-IP-URPF-MIB lets you specify a Unicast RPF drop-rate threshold on interfaces of a managed device that will send an SNMP notification when the threshold is exceeded. The MIB includes objects for specifying global and per-interface drop counts and drop rates and a method to generate SNMP traps when the drop rate exceeds a configurable per-interface threshold.

Although you can configure some parameters globally, you must configure the CISCO-IP-URPF-MIB on individual interfaces.

Security Policy and Unicast RPF

When determining how to deploy Unicast Reverse Path Forwarding (RPF), consider the following points:

- Apply Unicast RPF at the downstream interface, away from the larger portion of the network, preferably at the edges of your network. The further you apply Unicast RPF, the finer the granularity you have in mitigating address spoofing and in identifying sources of spoofed addresses. For example, applying Unicast RPF on an aggregation device helps to mitigate attacks from many downstream networks or clients and is simple to administer, but Unicast RPF does not help in identifying the source of the attack. Applying Unicast RPF at the network access server helps to limit the scope of the attack and trace the source of the attack. However, deploying Unicast RPF across many sites adds to the administration cost of operating a network.
- When you deploy Unicast RPF on many entities on a network (for example, across the Internet, intranet, and extranet resources), you have better chances of mitigating large-scale network disruptions throughout the Internet community, and of tracing the source of an attack.
- Unicast RPF does not inspect IP packets that are encapsulated in tunnels, such as the generic routing encapsulation (GRE), Layer 2 Tunneling Protocol (L2TP), or Point-to-Point Tunneling Protocol (PPTP). Configure Unicast RPF on a home gateway so that Unicast RPF processes network traffic only after tunneling and encryption layers are stripped off from the packets.

Ingress and Egress Filtering Policy for Unicast RPF

Note Unicast RPF with access control lists (ACLs) is not supported on the

Unicast RPF can be more effective at mitigating spoofing attacks when combined with a policy of ingress and egress filtering by using ACLs.

Ingress filtering applies filters to traffic that is received at a network interface from either internal or external networks. With ingress filtering, packets that arrive from other networks or the Internet and that have a source address that matches a local network or private or broadcast addresses are dropped. For example, in ISP environments, ingress filtering can be applied to traffic that is received at a device from either a client (customer) or the Internet.

Egress filtering applies filters to the traffic that exits a network interface (the sending interface). By filtering packets on devices that connect your network to the Internet or to other networks, you can permit only packets with valid source IP addresses to leave your network.

For more information on network filtering, refer to RFC 2267, *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing*.

Where to Use Unicast Reverse Path Forwarding

Unicast RPF can be used in any “single-homed” environment where there is essentially only one access point out of the network, which means that there is only one upstream connection to the network. Networks having one access point offer the best example of symmetric routing, which means that the interface where a packet enters the network is also the best return path to the source of the IP packet. Unicast RPF is best used at the network perimeter for Internet, intranet, or extranet environments, or in ISP environments for customer network terminations.

Routing Table Requirements

Unicast Reverse Path Forwarding (RPF) uses the routing information in Cisco Express Forwarding tables for routing traffic. The amount of routing information that must be available in Cisco Express Forwarding tables depends on the device where Unicast RPF is configured and the functions the device performs in the network. For example, in an ISP environment where a device is a leased-line aggregation device for customers, the information about static routes that are redistributed into the Interior Gateway Protocol (IGP) or Internal Border Gateway Protocol (IBGP) (depending on which technique is used in the network) is required in the routing table. Because Unicast RPF is configured on customer interfaces, only minimal routing information is required. If a single-homed ISP configures Unicast RPF on the gateway to the Internet, the full Internet routing table information is required by Unicast RPF to help protect the ISP from external denial of service (DoS) attacks that use addresses that are not in the Internet routing table.

Where Not to Use Unicast Reverse Path Forwarding

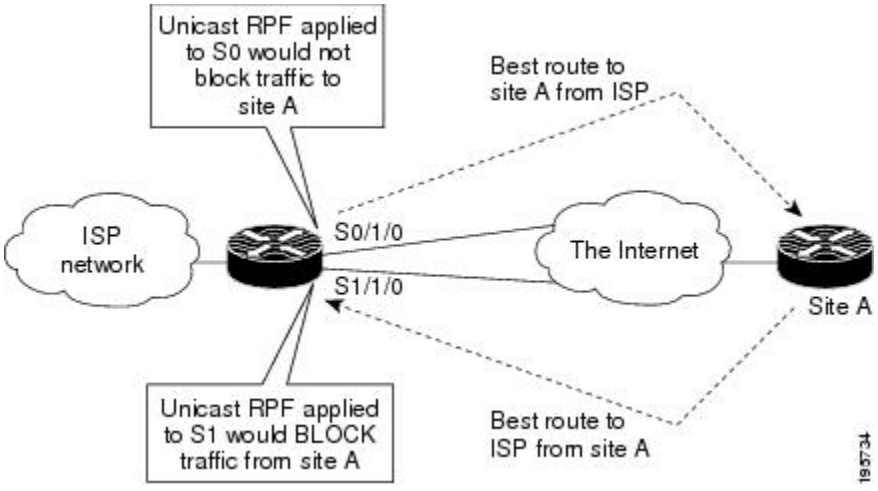
Do not use unicast RPF on interfaces that are internal to a network. Internal interfaces are likely to have routing asymmetry (see the figure below), which means that there can be multiple routes to the source of a packet. Unicast RPF is applied only where there is a natural or configured symmetry.

For example, devices at the edge of an ISP network are more likely to have symmetrical reverse paths than devices that are in the core of an ISP network. The best forwarding path to forward packets from devices that are at the core of an ISP network may not be the best forwarding path that is selected for packets that are returned to the device.

We recommend that you do not apply Unicast RPF where there is a chance of asymmetric routing, unless you configure access control lists (ACLs) to allow the device to accept incoming packets. ACLs permit the use of Unicast RPF when packets arrive through specific, less-optimal asymmetric input paths.

The figure below illustrates how Unicast RPF can block legitimate traffic in an asymmetric routing environment.

Figure 51: Unicast RPF Blocking Legitimate Traffic in an Asymmetric Routing Environment



Unicast Reverse Path Forwarding with BOOTP and DHCP

Unicast RPF allows packets with 0.0.0.0 as the source IP address and 255.255.255.255 as the destination IP address to pass through a network to enable Bootstrap Protocol (BOOTP) and DHCP functions to work properly when Unicast RPF is configured.

How to Configure Unicast Reverse Path Forwarding

The following section provide configuration information about unicast reverse path forwarding.

Configuring Unicast Reverse Path Forwarding

Before you begin

To use Unicast Reverse Path Forwarding, you must configure a device for Cisco Express Forwarding switching or distributed Cisco Express Forwarding switching. If Cisco Express Forwarding is not enabled globally on a device, Unicast RPF will not work on that device. If Cisco Express Forwarding is running on a device, individual interfaces on the device can be configured with other switching modes. Unicast RPF is an input-side function that is enabled on an interface or subinterface that supports any type of encapsulation, and Unicast RPF operates on IP packets that are received by the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip cef distributed** **Example:** `Device(config)# ip cef distributed`	Enables Cisco Express Forwarding or distributed Cisco Express Forwarding on a device.
Step 4	**interface** *slot/subslot/port* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Selects the input interface on which you want to apply Unicast Reverse Path Forwarding and enters interface configuration mode. The interface that is configured is the receiving interface, which allows Unicast RPF to verify the best return path before forwarding a packet to the next destination.
Step 5	**ip verify unicast reverse-path** *list*	Enables Unicast RPF on the interface.

	Command or Action	Purpose
	Example: `Device(config-if)# ip verify unicast reverse-path 197`	• Use the *list* argument to identify an access list. If the access list denies network access, spoofed packets are dropped at the interface. If the access list permits network access, spoofed packets are forwarded to the destination address. Forwarded packets are counted in the interface statistics. If the access list includes the logging option, information about the spoofed packets is logged to the log server. • Repeat this step for each access list that you want specify
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.

Troubleshooting Tips

HSRP Failure

The failure to disable Unicast RPF before disabling Cisco Express Forwarding can cause a Hot Standby Router Protocol (HSRP) failure. If you want to disable Cisco Express Forwarding on a device, you must first disable Unicast RPF.

Monitoring and Maintaining Unicast Reverse Path Forwarding

This section describes commands used to monitor and maintain unicast RPF.

Command	Purpose
`Device# show ip traffic`	Displays global router statistics about Unicast RPF drops and suppressed drops.
`Device# show ip interface type`	Displays per-interface statistics about Unicast RPF drops and suppressed drops.
`Device# show access-lists`	Displays the number of matches to a specific ACL.
`Device(config-if)# no ip verify unicast reverse-path list`	Disables Unicast RPF at the interface. Use the *list* option to disable Unicast RPF for a specific ACL at the interface.

Caution To disable CEF, you must first disable Unicast RPF. Failure to disable Unicast RPF before disabling CEF can cause HSRP failure. If you want to disable CEF on the router, you must first disable Unicast RPF.

Unicast RPF counts the number of packets dropped or suppressed because of malformed or forged source addresses. Unicast RPF counts dropped or forwarded packets that include the following global and per-interface information:

- Global Unicast RPF drops
- Per-interface Unicast RPF drops
- Per-interface Unicast RPF suppressed drops

The **show ip traffic** command shows the total number (global count) of dropped or suppressed packets for all interfaces on the router. The Unicast RPF drop count is included in the IP statistics section.

```
Device# show ip traffic

IP statistics:
  Rcvd:  1471590 total, 887368 local destination
         0 format errors, 0 checksum errors, 301274 bad hop count
         0 unknown protocol, 0 not a gateway
         0 security failures, 0 bad options, 0 with options
  Opts:  0 end, 0 nop, 0 basic security, 0 loose source route
         0 timestamp, 0 extended security, 0 record route
         0 stream ID, 0 strict source route, 0 alert, 0 other
  Frags: 0 reassembled, 0 timeouts, 0 couldn't reassemble
         0 fragmented, 0 couldn't fragment
  Bcast: 205233 received, 0 sent
  Mcast: 463292 received, 462118 sent
  Sent:  990158 generated, 282938 forwarded
  ! The second line below ("0 unicast RPF") displays Unicast RPF packet dropping
information.
  Drop:  3 encapsulation failed, 0 unresolved, 0 no adjacency
         0 no route, 0 unicast RPF, 0 forced drop
```

A nonzero value for the count of dropped or suppressed packets can mean one of two things:

- Unicast RPF is dropping or suppressing packets that have a bad source address (normal operation).
- Unicast RPF is dropping or suppressing legitimate packets because the route is misconfigured to use Unicast RPF in environments where asymmetric routing exists; that is, where multiple paths can exist as the best return path for a source address.

The **show ip interface** command shows the total of dropped or suppressed packets at a specific interface. If Unicast RPF is configured to use a specific ACL, that ACL information is displayed along with the drop statistics.

```
Device> show ip interface gigabitethernet1/1

   Unicast RPF ACL 197
   1 unicast RPF drop
   1 unicast RPF suppressed drop
```

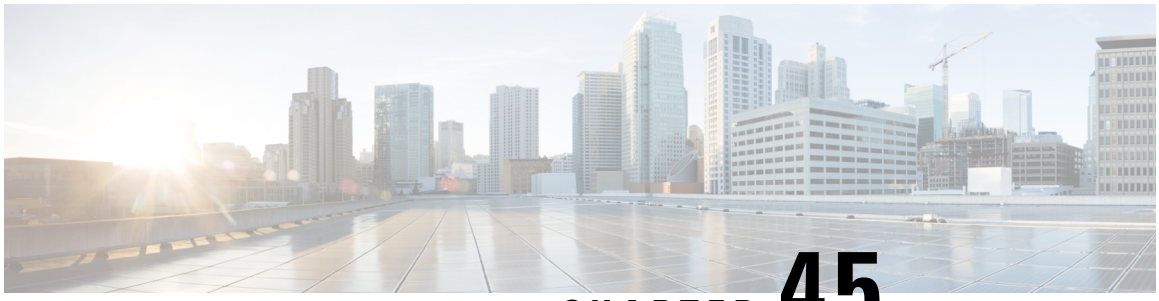
The **show access-lists** command displays the number of matches found for a specific entry in a specific access list.

```
Device> show access-lists

Extended IP access list 197
     deny ip 192.168.201.0 0.0.0.63 any log-input (1 match)
     permit ip 192.168.201.64 0.0.0.63 any log-input (1 match)
     deny ip 192.168.201.128 0.0.0.63 any log-input
     permit ip 192.168.201.192 0.0.0.63 any log-input
```

Example: Configuring Unicast RPF

```
Device# configure terminal
Device(config)# ip cef distributed
Device(config)# interface GigabitEthernet 1/2
Device(config-if)# description Connection to Upstream ISP
Device(config-if)# ip address 209.165.200.225 255.255.255.252
Device(config-if)# no ip redirects
Device(config-if)# no ip directed-broadcast
Device(config-if)# no ip proxy-arp
Device(config-if)# ip verify unicast reverse-path
```

```
Device# configure terminal
Device(config)# ip cef distributed
Device(config)# interface GigabitEthernet 1/2
Device(config-if)# description Connection to Upstream ISP
Device(config-if)# ip address 209.165.200.225 255.255.255.252
Device(config-if)# no ip redirects
Device(config-if)# no ip directed-broadcast
Device(config-if)# no ip proxy-arp
Device(config-if)# ip verify unicast source reachable-via rx
```



CHAPTER 45

Protocol-Independent Features

Distributed Cisco Express Forwarding and Load-Balancing Scheme for CEF Traffic

The following sections provide information about distributed Cisco express forwarding (CEF) and load-balancing scheme for CEF traffic.

Restrictions for Configuring a Load-Balancing Scheme for CEF Traffic

- You must globally configure load balancing on devicemembers in the same way.
- Per-packet load balancing for CEF traffic is not supported.

Information About Cisco Express Forwarding

Cisco Express Forwarding (CEF) is a Layer 3 IP switching technology used to optimize network performance. CEF implements an advanced IP look-up and forwarding algorithm to deliver maximum Layer 3 switching performance. CEF is less CPU-intensive than fast switching route caching, allowing more CPU processing power to be dedicated to packet forwarding. In dynamic networks, fast switching cache entries are frequently invalidated because of routing changes, which can cause traffic to be process switched using the routing table, instead of fast switched using the route cache. CEF and dCEF use the Forwarding Information Base (FIB) lookup table to perform destination-based switching of IP packets.

The two main components in CEF and dCEF are the distributed FIB and the distributed adjacency tables.

- The FIB is similar to a routing table or information base and maintains a mirror image of the forwarding information in the IP routing table. When routing or topology changes occur in the network, the IP routing table is updated, and those changes are reflected in the FIB. The FIB maintains next-hop address information based on the information in the IP routing table. Because the FIB contains all known routes that exist in the routing table, CEF eliminates route cache maintenance, is more efficient for switching traffic, and is not affected by traffic patterns.
- Nodes in the network are said to be adjacent if they can reach each other with a single hop across a link layer. CEF uses adjacency tables to prepend Layer 2 addressing information. The adjacency table maintains Layer 2 next-hop addresses for all FIB entries.

Because the switchuses Application Specific Integrated Circuits (ASICs) to achieve Gigabit-speed line rate IP traffic, CEF or dCEF forwarding applies only to the software-forwarding path, that is, traffic that is forwarded by the CPU.

CEF Load-Balancing Overview

CEF load balancing allows you to optimize resources by distributing traffic over multiple paths. CEF load balancing works based on a combination of source and destination packet information.

You can configure load balancing on a per-destination. Because load-balancing decisions are made on the outbound interface, load balancing must be configured on the outbound interface.

Per-Destination Load Balancing for CEF Traffic

Per-destination load balancing allows the device to use multiple paths to achieve load sharing across multiple source-destination host pairs. Packets for a given source-destination host pair are guaranteed to take the same path, even if multiple paths are available. Traffic streams destined for different pairs tend to take different paths.

Per-destination load balancing is enabled by default when you enable CEF. To use per-destination load balancing, you do not perform any additional tasks once CEF is enabled. Per-destination is the load-balancing method of choice for most situations.

Because per-destination load balancing depends on the statistical distribution of traffic, load sharing becomes more effective as the number of source-destination host pairs increases.

You can use per-destination load balancing to ensure that packets for a given host pair arrive in order. All packets intended for a certain host pair are routed over the same link (or links).

Load-Balancing Algorithms for CEF Traffic

The following load-balancing algorithms are provided for use with CEF traffic. Select a load-balancing algorithm with the **ip cef load-sharing algorithm** command.

- Original algorithm—The original load-balancing algorithm produces distortions in load sharing across multiple devices because the same algorithm was used on every device. Depending on your network environment, you should select the algorithm.
- Universal algorithm—The universal load-balancing algorithm allows each device on the network to make a different load sharing decision for each source-destination address pair, which resolves load-sharing imbalances. The device is set to perform universal load sharing by default.

How to Configure Cisco Express Forwarding

CEF or distributed CEF is enabled globally by default. If for some reason it is disabled, you can re-enable it by using the **ip cef** or **ip cef distributed** global configuration command.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**ip cef** **Example:** `Device(config)# ip cef`	Enables CEF operation on a non-stacking switch. Go to Step 4.
Step 3	**ip cef distributed** **Example:** `Device(config)# ip cef distributed`	Enables CEF operation on a switch.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 5	**ip route-cache cef** **Example:** `Device(config-if)# ip route-cache cef`	Enables CEF on the interface for software-forwarded traffic. **Note** The **ip route-cache cef** command is enabled by default and it cannot be disabled.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 7	**show ip cef** **Example:** `Device# show ip cef`	Displays the CEF status on all interfaces.

	Command or Action	Purpose
Step 8	**show cef linecard** [**detail**] **Example:** `Device# show cef linecard detail`	(Optional) Displays CEF-related interface information on a switch.
Step 9	**show cef interface** [*interface-id*] **Example:** `Device# show cef interface gigabitethernet 1/1`	Displays detailed CEF information for all interfaces or the specified interface.
Step 10	**show adjacency** **Example:** `Device# show adjacency`	Displays CEF adjacency table information.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

How to Configure a Load-Balancing for CEF Traffic

The following sections provide information on configuring load-balancing for CEF traffic.

Enabling or Disabling CEF Per-Destination Load Balancing

To enable or disable CEF per-destination load balancing, perform the following procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enters global configuration mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *interface-id* **Example:** `Device(config-if)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the Layer 3 interface to configure.
Step 4	[**no**] **ip load-sharing per-destination** **Example:** `Device(config-if)# ip load-sharing per-destination`	Enables per-destination load balancing for CEF on the interface. The **no ip load-sharing per-destination** command disables per-destination load balancing for CEF on the interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Selecting a Tunnel Load-Balancing Algorithm for CEF Traffic

Select the tunnel algorithm when your network environment contains only a few source and destination pairs. The device is set to perform universal load sharing by default.

To select a tunnel load-balancing algorithm for CEF traffic, perform the following procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enters global configuration mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip cef load-sharing algorithm** {**original** | **universal** [*id*] } **Example:** `Device(config)# ip cef load-sharing algorithm universal`	Selects a CEF load-balancing algorithm. • The **original** keyword sets the load-balancing algorithm to the original algorithm, based on a source IP and destination IP hash. • The **universal** keyword sets the load-balancing algorithm to one that uses

	Command or Action	Purpose
		a source IP, destination IP, Layer 3 Protocol, Layer 4 source port, Layer 4 destination port and IPv6 flow label (for IPv6 traffic). • The *id* argument is a fixed identifier.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Example: Enabling or Disabling CEF Per-Destination Load Balancing

Per-destination load balancing is enabled by default when you enable CEF. The following example shows how to disable per-destination load balancing:

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet1/1
Device(config-if)# no ip load-sharing per-destination
Device(config-if)# end
```

Number of Equal-Cost Routing Paths

The following sections provide information about number of equal-cost routing paths.

Information About Equal-Cost Routing Paths

When a router has two or more routes to the same network with the same metrics, these routes can be thought of as having an equal cost. The term parallel path is another way to see occurrences of equal-cost routes in a routing table. If a router has two or more equal-cost paths to a network, it can use them concurrently. Parallel paths provide redundancy in case of a circuit failure and also enable a router to load balance packets over the available paths for more efficient use of available bandwidth.

Even though the router automatically learns about and configures equal-cost routes, you can control the maximum number of parallel paths supported by an IP routing protocol in its routing table. Although the switch software allows a maximum of 32 equal-cost routes, the switch hardware will never use more than 16 paths per route.

How to Configure Equal-Cost Routing Paths

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router** {**rip** | **ospf** | **eigrp**} **Example:** `Device(config)# router eigrp`	Enters router configuration mode.
Step 4	**maximum-paths** *maximum* **Example:** `Device(config-router)# maximum-paths 2`	Sets the maximum number of parallel paths for the protocol routing table. The range is from 1 to 16; the default is 4 for most IP routing protocols, but only 1 for BGP.
Step 5	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.
Step 6	**show ip protocols** **Example:** `Device# show ip protocols`	Verifies the setting in the *Maximum path* field.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Static Unicast Routes

The following sections provide information about static unicast routes.

Information About Static Unicast Routes

Static unicast routes are user-defined routes that cause packets moving between a source and a destination to take a specified path. Static routes can be important if the router cannot build a route to a particular destination and are useful for specifying a gateway of last resort to which all unroutable packets are sent.

The switch retains static routes until you remove them. However, you can override static routes with dynamic routing information by assigning administrative distance values. Each dynamic routing protocol has a default administrative distance, as listed in Table 41-16. If you want a static route to be overridden by information from a dynamic routing protocol, set the administrative distance of the static route higher than that of the dynamic protocol.

Table 55: Dynamic Routing Protocol Default Administrative Distances

Route Source	Default Distance
Connected interface	0
Static route	1
Enhanced IRGP summary route	5
Internal Enhanced IGRP	90
IGRP	100
OSPF	110
Unknown	225

Static routes that point to an interface are advertised through RIP, IGRP, and other dynamic routing protocols, whether or not static **redistribute** router configuration commands were specified for those routing protocols. These static routes are advertised because static routes that point to an interface are considered in the routing table to be connected and hence lose their static nature. However, if you define a static route to an interface that is not one of the networks defined in a network command, no dynamic routing protocols advertise the route unless a **redistribute** static command is specified for these protocols.

When an interface goes down, all static routes through that interface are removed from the IP routing table. When the software can no longer find a valid next hop for the address specified as the forwarding router's address in a static route, the static route is also removed from the IP routing table.

Configuring Static Unicast Routes

Static unicast routes are user-defined routes that cause packets moving between a source and a destination to take a specified path. Static routes can be important if the router cannot build a route to a particular destination and are useful for specifying a gateway of last resort to which all unroutable packets are sent.

Follow these steps to configure a static route:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip route prefix mask** {*address* | *interface*} [*distance*] **Example:** `Device(config)# ip route prefix mask gigabitethernet 1/4`	Establish a static route.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show ip route** **Example:** `Device# show ip route`	Displays the current state of the routing table to verify the configuration.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

Use the **no ip route** *prefix mask* {*address*| *interface*} global configuration command to remove a static route. The device retains static routes until you remove them.

Default Routes and Networks

The following sections provides information about default routes and networks.

Information About Default Routes and Networks

A router might not be able to learn the routes to all other networks. To provide complete routing capability, you can use some routers as smart routers and give the remaining routers default routes to the smart router. (Smart routers have routing table information for the entire internetwork.) These default routes can be dynamically learned or can be configured in the individual routers. Most dynamic interior routing protocols include a mechanism for causing a smart router to generate dynamic default information that is then forwarded to other routers.

If a router has a directly connected interface to the specified default network, the dynamic routing protocols running on that device generate a default route. In RIP, it advertises the pseudonetwork 0.0.0.0.

A router that is generating the default for a network also might need a default of its own. One way a router can generate its own default is to specify a static route to the network 0.0.0.0 through the appropriate device.

When default information is passed through a dynamic routing protocol, no further configuration is required. The system periodically scans its routing table to choose the optimal default network as its default route. In IGRP networks, there might be several candidate networks for the system default. Cisco routers use administrative distance and metric information to set the default route or the gateway of last resort.

If dynamic default information is not being passed to the system, candidates for the default route are specified with the **ip default-network** global configuration command. If this network appears in the routing table from any source, it is flagged as a possible choice for the default route. If the router has no interface on the default network, but does have a path to it, the network is considered as a possible candidate, and the gateway to the best default path becomes the gateway of last resort.

How to Configure Default Routes and Networks

To configure default routes and networks, perform the following steps:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**ip default-network** *network number* **Example:** `Device(config)# ip default-network 1`	Specifies a default network.

	Command or Action	Purpose
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 4	**show ip route** **Example:** `Device# show ip route`	Displays the selected default route in the gateway of last resort display.
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Multiple Next Hops

The Routing Information Base (RIB) previously did not have a dedicated YANG model to retrieve the route information. Users had to use the IETF routing and routing-state models to query the next hops for a given route or prefix information from a device. These IETF models, only provided information about a single next-hop per route; even if the route had more than one next hop.

A new container is added under the *next-hop-options* choice node to retrieve all next-hops for a given route or prefix. Also, an *uptime* leaf node is added to provide the timestamp for each next hop.

When you query the routing-state model, if there are multiple next hops under a route, each next hop is listed with its index in the route.

The following is a sample RPC reply for a route with multiple next hops:

```
<rpc-reply xmlns="..." xmlns:nc="..." message-id="...">
  <data>
    <routing-state xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
      <routing-instance>
        <ribs>
          <rib>
            <routes>
              <route>
                <destination-prefix>172.16.1.0/24</destination-prefix>
                <route-preference>120</route-preference>
                <metric>1</metric>
                <next-hop>
                  <next-hop-list>
                    <next-hop>
                      <index>1</index>
                      <outgoing-interface>GigabitEthernet2</outgoing-interface>
                      <next-hop-address>10.1.1.2</next-hop-address>
                      <uptime>00:00:30</uptime>
                    </next-hop>
                    <next-hop>
                      <index>2</index>
                      <outgoing-interface>GigabitEthernet3</outgoing-interface>
```

```
                    <next-hop-address>10.2.1.2</next-hop-address>
                    <uptime>00:00:28</uptime>
                  </next-hop>
                </next-hop-list>
              </next-hop>
              <source-protocol>rip</source-protocol>
              <active/>
            </route>
          </routes>
        </rib>
      </ribs>
    </routing-instance>
  </routing-state>
 </data>
</rpc-reply>
```

When you query a route that only has a single next hop, the returned RPC will now have the next hop uptime (only for dynamic routing protocols).

The following is a sample RPC reply for a route with single next hop:

```
<rpc-reply xmlns="..." xmlns:nc="..." message-id="...">
  <data>
    <routing-state xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
      <routing-instance>
        <ribs>
          <rib>
            <routes>
              <route>
                <destination-prefix>172.16.1.0/24</destination-prefix>
                <route-preference>120</route-preference>
                <metric>1</metric>
                <next-hop>
                  <outgoing-interface>GigabitEthernet2</outgoing-interface>
                  <next-hop-address>10.1.1.2</next-hop-address>
                  <uptime>00:00:30</uptime>
                </next-hop>
                <source-protocol>rip</source-protocol>
                <active/>
              </route>
            </routes>
          </rib>
        </ribs>
      </routing-instance>
    </routing-state>
  </data>
</rpc-reply>
```

Route Maps to Redistribute Routing Information

The following sections provide information about route maps to redistribute routing information.

Information About Route Maps

The switch can run multiple routing protocols simultaneously, and it can redistribute information from one routing protocol to another. Redistributing information from one routing protocol to another applies to all supported IP-based routing protocols.

You can also conditionally control the redistribution of routes between routing domains by defining enhanced packet filters or route maps between the two domains. The **match** and **set** route-map configuration commands define the condition portion of a route map. The **match** command specifies that a criterion must be matched. The **set** command specifies an action to be taken if the routing update meets the conditions defined by the match command. Although redistribution is a protocol-independent feature, some of the **match** and **set** route-map configuration commands are specific to a particular protocol.

One or more **match** commands and one or more **set** commands follow a **route-map** command. If there are no **match** commands, everything matches. If there are no **set** commands, nothing is done, other than the match. Therefore, you need at least one **match** or **set** command.

Note A route map with no **set** route-map configuration commands is sent to the CPU, which causes high CPU utilization.

You can also identify route-map statements as **permit** or **deny**. If the statement is marked as a deny, the packets meeting the match criteria are sent back through the normal forwarding channels (destination-based routing). If the statement is marked as permit, set clauses are applied to packets meeting the match criteria. Packets that do not meet the match criteria are forwarded through the normal routing channel.

How to Configure a Route Map

Although each of Steps 3 through 14 in the following section is optional, you must enter at least one **match** route-map configuration command and one **set** route-map configuration command.

Note The keywords are the same as defined in the procedure to control the route distribution.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**route-map***map-tag* [**permit** | **deny**] [*sequence number*] **Example:** `Device(config)# route-map rip-to-ospf permit 4`	Defines any route maps used to control redistribution and enter route-map configuration mode. *map-tag*—A meaningful name for the route map. The **redistribute** router configuration command uses this name to reference this route map. Multiple route maps might share the same map tag name. (Optional) If **permit** is specified and the match criteria are met for this route map, the route is

	Command or Action	Purpose
		redistributed as controlled by the set actions. If **deny** is specified, the route is not redistributed. *sequence number* (Optional)— Number that indicates the position a new route map is to have in the list of route maps already configured with the same name.
Step 3	**match as-path** *path-list-number* **Example:** `Device(config-route-map)# match as-path 10`	Matches a BGP AS path access list.
Step 4	**match community-list** *community-list-number* [**exact**] **Example:** `Device(config-route-map)# match community-list 150`	Matches a BGP community list.
Step 5	**match ip address** {*access-list-number* | *access-list-name*} [*...access-list-number* | *...access-list-name*] **Example:** `Device(config-route-map)# match ip address 5 80`	Matches a standard access list by specifying the name or number. It can be an integer from 1 to 199.
Step 6	**match metric** metric-value **Example:** `Device(config-route-map)# match metric 2000`	Matches the specified route metric. The *metric-value* can be an EIGRP metric with a specified value from 0 to 4294967295.
Step 7	**match ip next-hop** {*access-list-number* | *access-list-name*} [*...access-list-number* | *...access-list-name*] **Example:** `Device(config-route-map)# match ip next-hop 8 45`	Matches a next-hop router address passed by one of the access lists specified (numbered from 1 to 199).
Step 8	**match tag** *tag value* [*...tag-value*] **Example:** `Device(config-route-map)# match tag 3500`	Matches the specified tag value in a list of one or more route tag values. Each can be an integer from 0 to 4294967295.

	Command or Action	Purpose
Step 9	**match interface***type number* [*...type-number*] **Example:** `Device(config-route-map)# match interface gigabitethernet 1/1`	Matches the specified next hop route out one of the specified interfaces.
Step 10	**match ip route-source** {*access-list-number* | *access-list-name*} [*...access-list-number* | *...access-list-name*] **Example:** `Device(config-route-map)# match ip route-source 10 30`	Matches the address specified by the specified advertised access lists.
Step 11	**match route-type** {**local** | **internal** | **external** [**type-1** | **type-2**]} **Example:** `Device(config-route-map)# match route-type local`	Matches the specified **route-type**: • **local**—Locally generated BGP routes. • **internal**—OSPF intra-area and interarea routes or EIGRP internal routes. • **external**—OSPF external routes (Type 1 or Type 2) or EIGRP external routes.
Step 12	**set dampening** *halflife reuse suppress max-suppress-time* **Example:** `Device(config-route-map)# set dampening 30 1500 10000 120`	Sets BGP route dampening factors.
Step 13	**set local-preference** *value* **Example:** `Device(config-route-map)# set local-preference 100`	Assigns a value to a local BGP path.
Step 14	**set origin** {**igp** | **egp** *as* | **incomplete**} **Example:** `Device(config-route-map)# set origin igp`	Sets the BGP origin code.
Step 15	**set as-path** {**tag** | **prepend** *as-path-string*} **Example:** `Device(config-route-map)# set as-path tag`	Modifies the BGP autonomous system path.

	Command or Action	Purpose
Step 16	**set level** {**level-1** | **level-2** | **level-1-2** | **stub-area** | **backbone**} **Example:** `Device(config-route-map)# set level level-1-2`	Sets the level for routes that are advertised into the specified area of the routing domain. The **stub-area** and **backbone** are OSPF NSSA and backbone areas.
Step 17	**set metric** *metric value* **Example:** `Device(config-route-map)# set metric 100`	Sets the metric value to give the redistributed routes (for EIGRP only). The *metric value* is an integer from -294967295 to 294967295.
Step 18	**set metric***bandwidth delay reliability loading mtu* **Example:** `Device(config-route-map)# set metric 10000 10 255 1 1500`	Sets the metric value to give the redistributed routes (for EIGRP only): • *bandwidth*—Metric value or IGRP bandwidth of the route in kilobits per second in the range 0 to 4294967295 • *delay*—Route delay in tens of microseconds in the range 0 to 4294967295. • *reliability*—Likelihood of successful packet transmission expressed as a number between 0 and 255, where 255 means 100 percent reliability and 0 means no reliability. • *loading*—Effective bandwidth of the route expressed as a number from 0 to 255 (255 is 100 percent loading). • *mtu*—Minimum maximum transmission unit (MTU) size of the route in bytes in the range 0 to 4294967295.
Step 19	**set metric-type** {**type-1** | **type-2**} **Example:** `Device(config-route-map)# set metric-type type-2`	Sets the OSPF external metric type for redistributed routes.
Step 20	**set metric-type internal** **Example:** `Device(config-route-map)# set metric-type internal`	Sets the multi-exit discriminator (MED) value on prefixes advertised to external BGP neighbor to match the IGP metric of the next hop.

	Command or Action	Purpose
Step 21	**set weight** *number* **Example:** `Device(config-route-map)# set weight 100`	Sets the BGP weight for the routing table. The value can be from 1 to 65535.
Step 22	**end** **Example:** `Device(config-route-map)# end`	Returns to privileged EXEC mode.
Step 23	**show route-map** **Example:** `Device# show route-map`	Displays all route maps configured or only the one specified to verify configuration.
Step 24	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

How to Control Route Distribution

Although each of Steps 3 through 14 in the following section is optional, you must enter at least one **match** route-map configuration command and one **set** route-map configuration command.

Note The keywords are the same as defined in the procedure to configure the route map for redistritbution.

The metrics of one routing protocol do not necessarily translate into the metrics of another. For example, the RIP metric is a hop count, and the IGRP metric is a combination of five qualities. In these situations, an artificial metric is assigned to the redistributed route. Uncontrolled exchanging of routing information between different routing protocols can create routing loops and seriously degrade network operation.

If you have not defined a default redistribution metric that replaces metric conversion, some automatic metric translations occur between routing protocols:

- RIP can automatically redistribute static routes. It assigns static routes a metric of 1 (directly connected).
- Any protocol can redistribute other routing protocols if a default mode is in effect.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**router** {**rip** | **ospf** | **eigrp**} **Example:** `Device(config)# router eigrp 10`	Enters router configuration mode.
Step 3	**redistribute** *protocol* [*process-id*] {**level-1** | **level-1-2** | **level-2**} [**metric** *metric-value*] [**metric-type** *type-value*] [**match internal** | **external** *type-value*] [**tag** *tag-value*] [**route-map** *map-tag*] [**weight** *weight*] [**subnets**] **Example:** `Device(config-router)# redistribute eigrp 1`	Redistributes routes from one routing protocol to another routing protocol. If no route-maps are specified, all routes are redistributed. If the keyword **route-map** is specified with no *map-tag*, no routes are distributed.
Step 4	**default-metric** *number* **Example:** `Device(config-router)# default-metric 1024`	Cause the current routing protocol to use the same metric value for all redistributed routes (RIP and OSPF).
Step 5	**default-metric** bandwidth delay reliability loading mtu **Example:** `Device(config-router)# default-metric 1000 100 250 100 1500`	Cause the EIGRP routing protocol to use the same metric value for all non-EIGRP redistributed routes.
Step 6	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.
Step 7	**show route-map** **Example:** `Device# show route-map`	Displays all route maps configured or only the one specified to verify configuration.

	Command or Action	Purpose
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Policy-Based Routing

Restrictions for Configuring Policy-based Routing

- Policy-based routing (PBR) is not supported to forward traffic into GRE tunnel. This applies to PBR applied on any interface and forwarding traffic into GRE tunnel (by means of PBR next-hop or default next-hop or set interface).
- PBR is not supported on GRE tunnel itself (applied under the GRE tunnel itself).
- PBR does not apply to fragmented traffic. Fragmented traffic will follow a normal routing path.
- PBR and Network Address Translation (NAT) are not supported on the same interface. PBR and NAT work together only if they are configured on different interfaces.

Information About Policy-Based Routing

You can use policy-based routing (PBR) to configure a defined policy for traffic flows. By using PBR, you can have more control over routing by reducing the reliance on routes derived from routing protocols. PBR can specify and implement routing policies that allow or deny paths based on:

- Identity of a particular end system
- Application
- Protocol

You can use PBR to provide equal-access and source-sensitive routing, routing based on interactive versus batch traffic, or routing based on dedicated links. For example, you could transfer stock records to a corporate office on a high-bandwidth, high-cost link for a short time while transmitting routine application data such as e-mail over a low-bandwidth, low-cost link.

With PBR, you classify traffic using access control lists (ACLs) and then make traffic go through a different path. PBR is applied to incoming packets. All packets received on an interface with PBR enabled are passed through route maps. Based on the criteria defined in the route maps, packets are forwarded (routed) to the appropriate next hop.

- Route map statement marked as permit is processed as follows:
  - A match command can match on length or multiple ACLs. A route map statement can contain multiple match commands. Logical or algorithm function is performed across all the match commands to reach a permit or deny decision.

For example:

match length A B

match ip address acl1 acl2

match ip address acl3

A packet is permitted if it is permitted by match length A B or acl1 or acl2 or acl3

- If the decision reached is permit, then the action specified by the set command is applied on the packet .
- If the decision reached is deny, then the PBR action (specified in the set command) is not applied. Instead the processing logic moves forward to look at the next route-map statement in the sequence (the statement with the next higher sequence number). If no next statement exists, PBR processing terminates, and the packet is routed using the default IP routing table.

You can use standard IP ACLs to specify match criteria for a source address or extended IP ACLs to specify match criteria based on an application, a protocol type, or an end station. The process proceeds through the route map until a match is found. If no match is found, normal destination-based routing occurs. There is an implicit deny at the end of the list of match statements.

If match clauses are satisfied, you can use a set clause to specify the IP addresses identifying the next hop router in the path.

Local PBR configuration supports setting DSCP marking for RADIUS packets generated for device administration purposes.

How to Configure PBR

- Multicast traffic is not policy-routed. PBR applies only to unicast traffic.
- You can enable PBR on a routed port or an SVI.
- The switch supports PBR based on match length.
- You can apply a policy route map to an EtherChannel port channel in Layer 3 mode, but you cannot apply a policy route map to a physical interface that is a member of the EtherChannel. If you try to do so, the command is rejected. When a policy route map is applied to a physical interface, that interface cannot become a member of an EtherChannel.
- When configuring match criteria in a route map, follow these guidelines:
  - Do not match ACLs that permit packets destined for a local address.
- VRF and PBR are mutually exclusive on a switch interface. You cannot enable VRF when PBR is enabled on an interface. The reverse is also true, you cannot enable PBR when VRF is enabled on an interface.
- Web Cache Communication Protocol (WCCP) and PBR are mutually exclusive on a switch interface. You cannot enable WCCP when PBR is enabled on an interface. The reverse is also true, you cannot enable PBR when WCCP is enabled on an interface.
- The number of hardware entries used by PBR depends on the route map itself, the ACLs used, and the order of the ACLs and route-map entries.
- PBR based on TOS, DSCP and IP Precedence are not supported.

- Set interface, set default next-hop and set default interface are not supported.
- **ip next-hop recursive** and **ip next-hop verify availability** features are not available and the next-hop should be directly connected.
- Policy-maps with no set actions are supported. Matching packets are routed normally.
- Policy-maps with no match clauses are supported. Set actions are applied to all packets.
- You can define a maximum of 128 IP policy route maps on the switch.

By default, PBR is disabled on the switch. To enable PBR, you must create a route map that specifies the match criteria and the resulting action. Then, you must enable PBR for that route map on an interface. All packets arriving on the specified interface matching the match clauses are subject to PBR.

Packets that are generated by the switch (CPU), or local packets, are not normally policy-routed. When you globally enable local PBR on the switch, all unicast packets that originate on the switch are subject to local PBR. The protocols that are supported for local PBR are NTP, DNS, MSDP, SYSLOG and TFTP. Local PBR is disabled by default.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *map-tag* [**permit**] [*sequence number*] **Example:** `Device(config)# route-map pbr-map permit`	Defines route maps that are used to control where packets are output, and enters route-map configuration mode. • *map-tag* — A meaningful name for the route map. The **ip policy route-map** interface configuration command uses this name to reference the route map. Multiple route-map statements with the same map tag define a single route map. • (Optional) **permit** — If **permit** is specified and the match criteria are met for this route map, the route is policy routed as defined by the set actions. • (Optional) *sequence number* — The sequence number shows the position of the route-map statement in the given route map.

	Command or Action	Purpose
Step 4	**match ip address** {*access-list-number* | *access-list-name*} [*access-list-number* |...*access-list-name*] **Example:** `Device(config-route-map)# match ip address 110 140`	Matches the source and destination IP addresses that are permitted by one or more standard or extended access lists. ACLs can match on more than one source and destination IP address. If you do not specify a **match** command, the route map is applicable to all packets.
Step 5	**match length min max** **Example:** `Device(config-route-map)# match length 64 1500`	Matches the length of the packet.
Step 6	**set ip next-hop** *ip-address* [*...ip-address*] **Example:** `Device(config-route-map)# set ip next-hop 10.1.6.2`	Specifies the action to be taken on the packets that match the criteria. Sets next hop to which to route the packet (the next hop must be adjacent).
Step 7	**exit** **Example:** `Device(config-route-map)# exit`	Returns to global configuration mode.
Step 8	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the interface to be configured.
Step 9	**ip policy route-map** *map-tag* **Example:** `Device(config-if)# ip policy route-map pbr-map`	Enables PBR on a Layer 3 interface, and identify the route map to use. You can configure only one route map on an interface. However, you can have multiple route map entries with different sequence numbers. These entries are evaluated in the order of sequence number until the first match. If there is no match, packets are routed as usual.
Step 10	**ip route-cache policy** **Example:** `Device(config-if)# ip route-cache policy`	(Optional) Enables fast-switching PBR. You must enable PBR before enabling fast-switching PBR.
Step 11	**exit** **Example:** `Device(config-if)# exit`	Returns to global configuration mode.
Step 12	**ip local policy route-map** *map-tag* **Example:**	(Optional) Enables local PBR to perform policy-based routing on packets originating at

	Command or Action	Purpose
	`Device(config)# ip local policy route-map local-pbr`	the switch. This applies to packets generated by the switch, and not to incoming packets.
Step 13	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 14	**show route-map** [*map-name*] **Example:** `Device# show route-map`	(Optional) Displays all the route maps configured or only the one specified to verify configuration.
Step 15	**show ip policy** **Example:** `Device# show ip policy`	(Optional) Displays policy route maps attached to the interface.
Step 16	**show ip local policy** **Example:** `Device# show ip local policy`	(Optional) Displays whether or not local policy routing is enabled and, if so, the route map being used.

Filtering Routing Information

Note When routes are redistributed between OSPF processes, no OSPF metrics are preserved.

Setting Passive Interfaces

To prevent other routers on a local network from dynamically learning about routes, you can use the **passive-interface** router configuration command to keep routing update messages from being sent through a router interface. When you use this command in the OSPF protocol, the interface address you specify as passive appears as a stub network in the OSPF domain. OSPF routing information is neither sent nor received through the specified router interface.

In networks with many interfaces, to avoid having to manually set them as passive, you can set all interfaces to be passive by default by using the **passive-interface default** router configuration command and manually setting interfaces where adjacencies are desired.

Use a network monitoring privileged EXEC command such as **show ip ospf interface** to verify the interfaces that you enabled as passive, or use the **show ip interface** privileged EXEC command to verify the interfaces that you enabled as active.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**router** {**rip** | **ospf** | **eigrp**} **Example:** `Device(config)# router ospf`	Enters router configuration mode.
Step 3	**passive-interface** *interface-id* **Example:** `Device(config-router)# passive-interface gigabitethernet 1/1`	Suppresses sending routing updates through the specified Layer 3 interface.
Step 4	**passive-interface default** **Example:** `Device(config-router)# passive-interface default`	(Optional) Sets all interfaces as passive by default.
Step 5	**no passive-interface** *interface type* **Example:** `Device(config-router)# no passive-interface gigabitethernet1/3 gigabitethernet 1/5`	(Optional) Activates only those interfaces that need to have adjacencies sent.
Step 6	**network** *network-address* **Example:** `Device(config-router)# network 10.1.1.1`	(Optional) Specifies the list of networks for the routing process. The *network-address* is an IP address.
Step 7	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Controlling Advertising and Processing in Routing Updates

You can use the **distribute-list** router configuration command with access control lists to suppress routes from being advertised in routing updates and to prevent other routers from learning one or more routes. When used in OSPF, this feature applies to only external routes, and you cannot specify an interface name.

You can also use a **distribute-list** router configuration command to avoid processing certain routes listed in incoming updates. (This feature does not apply to OSPF.)

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router** {**rip** | **eigrp**} **Example:** `Device(config)# router eigrp 10`	Enters router configuration mode.
Step 4	**distribute-list** {*access-list-number* | *access-list-name*} **out** [*interface-name* | *routing process* | *autonomous-system-number*] **Example:** `Device(config-router)# distribute 120 out gigabitethernet 1/7`	Permits or denies routes from being advertised in routing updates, depending upon the action listed in the access list.
Step 5	**distribute-list** {*access-list-number* | *access-list-name*} **in** [*type-number*] **Example:** `Device(config-router)# distribute-list 125 in`	Suppresses processing in routes listed in updates.
Step 6	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Filtering Sources of Routing Information

Because some routing information might be more accurate than others, you can use filtering to prioritize information coming from different sources. An administrative distance is a rating of the trustworthiness of a routing information source, such as a router or group of routers. In a large network, some routing protocols can be more reliable than others. By specifying administrative distance values, you enable the router to intelligently discriminate between sources of routing information. The router always picks the route whose routing protocol has the lowest administrative distance.

Because each network has its own requirements, there are no general guidelines for assigning administrative distances.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router** {**rip** | **ospf** | **eigrp**} **Example:** `Device(config)# router eigrp 10`	Enters router configuration mode.
Step 4	**distance** *weight* {*ip-address* {*ip-address mask*}} [*ip access list*] **Example:** `Device(config-router)# distance 50 10.1.5.1`	Defines an administrative distance. *weight*—The administrative distance as an integer from 10 to 255. Used alone, *weight* specifies a default administrative distance that is used when no other specification exists for a routing information source. Routes with a distance of 255 are not installed in the routing table.

	Command or Action	Purpose
		(Optional) *ip access list*—An IP standard or extended access list to be applied to incoming routing updates.
Step 5	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.
Step 6	**show ip protocols** **Example:** `Device# show ip protocols`	Displays the default administrative distance for a specified routing process.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Managing Authentication Keys

Key management is a method of controlling authentication keys used by routing protocols. Not all protocols can use key management. Authentication keys are available for EIGRP and RIP Version 2.

Prerequisites

Before you manage authentication keys, you must enable authentication. See the appropriate protocol section to see how to enable authentication for that protocol. To manage authentication keys, define a key chain, identify the keys that belong to the key chain, and specify how long each key is valid. Each key has its own key identifier (specified with the **key** *number* key chain configuration command), which is stored locally. The combination of the key identifier and the interface associated with the message uniquely identifies the authentication algorithm and Message Digest 5 (MD5) authentication key in use.

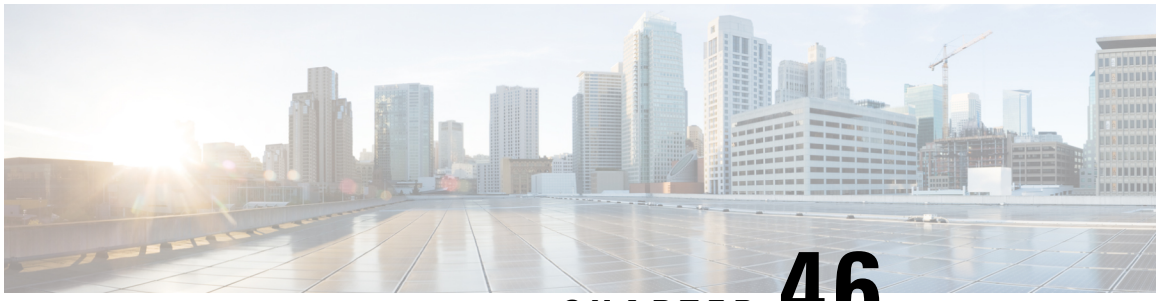
How to Configure Authentication Keys

You can configure multiple keys with life times. Only one authentication packet is sent, regardless of how many valid keys exist. The software examines the key numbers in order from lowest to highest, and uses the first valid key it encounters. The lifetimes allow for overlap during key changes. Note that the router must know these lifetimes.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**key chain** *name-of-chain* **Example:** `Device(config)# key chain key10`	Identifies a key chain, and enter key chain configuration mode.
Step 3	**key** *number* **Example:** `Device(config-keychain)# key 2000`	Identifies the key number. The range is 0 to 2147483647.
Step 4	**key-string** *text* **Example:** `Device(config-keychain)# Room 20, 10th floor`	Identifies the key string. The string can contain from 1 to 80 uppercase and lowercase alphanumeric characters, but the first character cannot be a number.
Step 5	**accept-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*} **Example:** `Device(config-keychain)# accept-lifetime 12:30:00 Jan 25 1009 infinite`	(Optional) Specifies the time period during which the key can be received. The *start-time* and *end-time* syntax can be either *hh:mm:ss Month date year* or *hh:mm:ss date Month year*. The default is forever with the default *start-time* and the earliest acceptable date as January 1, 1993. The default *end-time* and **duration** is **infinite**.
Step 6	**send-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*} **Example:** `Device(config-keychain)# accept-lifetime 23:30:00 Jan 25 1019 infinite`	(Optional) Specifies the time period during which the key can be sent. The *start-time* and *end-time* syntax can be either *hh:mm:ss Month date year* or *hh:mm:ss date Month year*. The default is forever with the default *start-time* and the earliest acceptable date as January 1, 1993. The default *end-time* and **duration** is **infinite**.
Step 7	**end** **Example:** `Device(config-keychain)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 8	**show key chain** **Example:** `Device# show key chain`	Displays authentication key information.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.



CHAPTER 46

Configuring Generic Routing Encapsulation(GRE) Tunnel IP Source and Destination VRF Membership

Restrictions for GRE Tunnel IP Source and Destination VRF Membership

- Both ends of the tunnel must reside within the same VRF.
- The VRF associated with the tunnel vrf command is the same as the VRF associated with the physical interface over which the tunnel sends packets (outer IP packet routing).
- The VRF associated with the tunnel by using the ip vrf forwarding command is the VRF that the packets are to be forwarded in as the packets exit the tunnel (inner IP packet routing).
- The feature does not support the fragmentation of multicast packets passing through a multicast tunnel.
- The feature does not support the ISIS (Intermediate System to intermediate system) protocol.
- Keepalive is not supported on VRF aware GRE tunnels.

Information About GRE Tunnel IP Source and Destination VRF Membership

This feature allows you to configure the source and destination of a tunnel to belong to any Virtual Private Network (VPN) routing and forwarding (VRF) table. A VRF table stores routing data for each VPN. The VRF table defines the VPN membership of a customer site attached to the network access server (NAS). Each

VRF table comprises an IP routing table, a derived Cisco Express Forwarding (CEF) table, and guidelines and routing protocol parameters that control the information that is included in the routing table.

Previously, GRE IP tunnels required the IP tunnel destination to be in the global routing table. The implementation of this feature allows you to configure a tunnel source and destination to belong to any VRF. As with existing GRE tunnels, the tunnel becomes disabled if no route to the tunnel destination is defined.

How to Configure GRE Tunnel IP Source and Destination VRF Membership

Follow these steps to configure GRE Tunnel IP Source and Destination VRF Membership:

Procedure

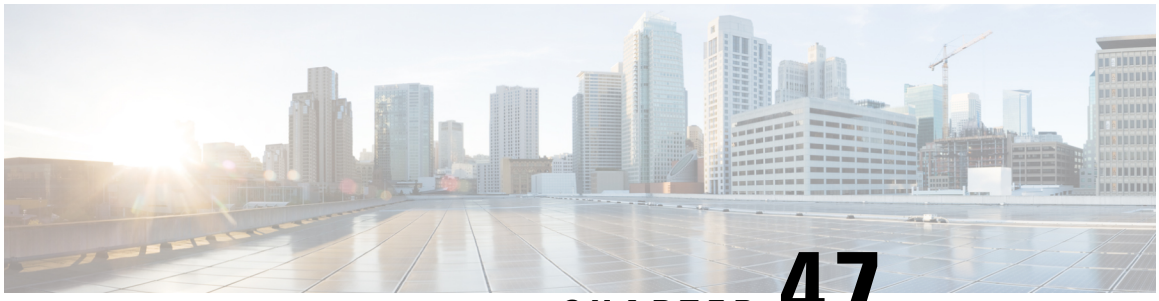
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel***number* **Example:** `Device(config)#interface tunnel 0`	Enters interface configuration mode for the specified interface. • *number* is the number associated with the tunnel interface.
Step 4	**ip vrf forwarding***vrf-name* **Example:** `Device(config-if)#ip vrf forwarding green`	Associates a virtual private network (VPN) routing and forwarding (VRF) instance with an interface or subinterface. • *vrf-name* is the name assigned to a VRF.
Step 5	**ip address***ip-address subnet-mask* **Example:** `Device(config-if)#ip address 10.7.7.7 255.255.255.255`	Specifies the interface IP address and subnet mask. • *ip-address* specifies the IP address of the interface. • *subnet-mask* specifies the subnet mask of the interface.
Step 6	**tunnel source** {*ip-address* | *type number*} **Example:** `Device(config-if)#tunnel source loop 0`	Specifies the source of the tunnel interface. • *ip-address* specifies the IP address to use as the source address for packets in the tunnel.

	Command or Action	Purpose
		• *type* specifies the interface type (for example, serial). • *number* specifies the port, connector, or interface card number. The numbers are assigned at the factory at the time of installation or when added to a system, and can be displayed using the show interfaces command.
Step 7	**tunnel destination** {*hostname* | *ip-address*} **Example:** `Device(config-if)#tunnel destination 10.5.5.5`	Defines the tunnel destination. • *hostname* specifies the name of the host destination. • *ip-address* specifies the IP address of the host destination.
Step 8	**tunnel vrf***vrf-name* **Example:** `Device(config-if)#tunnel vrf finance1`	Associates a VPN routing and forwarding (VRF) instance with a specific tunnel destination. • *vrf-name* is the name assigned to a VRF.

Configuration Example for GRE Tunnel IP Source and Destination VRF Membership

In this example, packets received on interface e0 using VRF green are forwarded out of the tunnel through interface e1 using VRF blue.

```
ip vrf blue rd 1:1

ip vrf green rd 1:2

interface loop0
ip vrf forwarding blue
ip address 10.7.7.7 255.255.255.255

interface tunnel0
ip vrf forwarding green
ip address 10.3.3.3 255.255.255.0 tunnel source loop 0
tunnel destination 10.5.5.5 tunnel vrf blue

interface GigabitEthernet 1/1
ip vrf forwarding green
ip address 10.1.1.1 255.255.255.0

interface GigabitEthernet 1/2
ip vrf forwarding blue
ip address 10.2.2.2 255.255.255.0

ip route vrf blue 10.5.5.5 255.255.255.0 GigabitEthernet 1/1
```



CHAPTER 47

IP Addressing Services Overview

This section provides information about IP Addressing Services.

Understanding IPv6

IPv4 users can move to IPv6 and receive services such as end-to-end security, quality of service (QoS), and globally unique addresses. The IPv6 address space reduces the need for private addresses and Network Address Translation (NAT) processing by border routers at network edges.

For information about how Cisco Systems implements IPv6, go to Networking Software (IOS & NX-OS)

For information about IPv6 and other features in this chapter

- See the *Cisco IOS IPv6 Configuration Library*.
- Use the Search field on Cisco.com to locate the Cisco IOS software documentation. For example, if you want information about static routes, you can enter *Implementing Static Routes for IPv6* in the search field to learn about static routes.

IPv6 Addresses

The switch supports only IPv6 unicast addresses. It does not support site-local unicast addresses, or anycast addresses.

The IPv6 128-bit addresses are represented as a series of eight 16-bit hexadecimal fields separated by colons in the format: n:n:n:n:n:n:n:n. This is an example of an IPv6 address:

2031:0000:130F:0000:0000:09C0:080F:130B

For easier implementation, leading zeros in each field are optional. This is the same address without leading zeros:

2031:0:130F:0:0:9C0:80F:130B

You can also use two colons (::) to represent successive hexadecimal fields of zeros, but you can use this short version only once in each address:

2031:0:130F::09C0:080F:130B

For more information about IPv6 address formats, address types, and the IPv6 packet header, see the IPv6 Addressing and Basic Connectivity Configuration Guide of *Cisco IOS IPv6 Configuration Library* on Cisco.com.

- IPv6 Address Formats
- IPv6 Address Type: Multicast
- IPv6 Address Output Display
- Simplified IPv6 Packet Header

128-Bit Wide Unicast Addresses

The switch supports aggregatable global unicast addresses and link-local unicast addresses. It does not support site-local unicast addresses.

- Aggregatable global unicast addresses are IPv6 addresses from the aggregatable global unicast prefix. The address structure enables strict aggregation of routing prefixes and limits the number of routing table entries in the global routing table. These addresses are used on links that are aggregated through organizations and eventually to the Internet service provider.

  These addresses are defined by a global routing prefix, a subnet ID, and an interface ID. Current global unicast address allocation uses the range of addresses that start with binary value 001 (2000::/3). Addresses with a prefix of 2000::/3(001) through E000::/3(111) must have 64-bit interface identifiers in the extended unique identifier (EUI)-64 format.

- Link local unicast addresses can be automatically configured on any interface by using the link-local prefix FE80::/10(1111 1110 10) and the interface identifier in the modified EUI format. Link-local addresses are used in the neighbor discovery protocol (NDP) and the stateless autoconfiguration process. Nodes on a local link use link-local addresses and do not require globally unique addresses to communicate. IPv6 routers do not forward packets with link-local source or destination addresses to other links.

For more information, see the section about IPv6 unicast addresses in the “Implementing IPv6 Addressing and Basic Connectivity” chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

DNS for IPv6

IPv6 supports Domain Name System (DNS) record types in the DNS name-to-address and address-to-name lookup processes. The DNS AAAA resource record types support IPv6 addresses and are equivalent to an A address record in IPv4. The switch supports DNS resolution for IPv4 and IPv6.

IPv6 Stateless Autoconfiguration and Duplicate Address Detection

The switch uses stateless autoconfiguration to manage link, subnet, and site addressing changes, such as management of host and mobile IP addresses. A host autonomously configures its own link-local address, and booting nodes send router solicitations to request router advertisements for configuring interfaces.

An autoconfigured IPv6 address contains interface identifiers that are not part of the reserved interface identifiers range specified in RFC5453.

For more information about autoconfiguration and duplicate address detection, see the Implementing IPv6 Addressing and Basic Connectivity chapter of *Cisco IOS IPv6 Configuration Library* on Cisco.com.

IPv6 Applications

The switch has IPv6 support for these applications:

- Ping, traceroute, Telnet, and TFTP
- FTP
- Secure Shell (SSH) over an IPv6 transport
- HTTP server access over IPv6 transport
- DNS resolver for AAAA over IPv4 transport
- Cisco Discovery Protocol (CDP) support for IPv6 addresses

For more information about managing these applications, see the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

DHCP for IPv6 Address Assignment

DHCPv6 enables DHCP servers to pass configuration parameters, such as IPv6 network addresses, to IPv6 clients. The address assignment feature manages non-duplicate address assignment in the correct prefix based on the network where the host is connected. Assigned addresses can be from one or multiple prefix pools. Additional options, such as default domain and DNS name-server address, can be passed back to the client. Address pools can be assigned for use on a specific interface, on multiple interfaces, or the server can automatically find the appropriate pool.

For configuring DHCP for IPv6, see the *Configuring DHCP for IPv6 Address Assignment* section.

For more information about configuring the DHCPv6 client, server, or relay agent functions, see the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

HTTP(S) Over IPv6

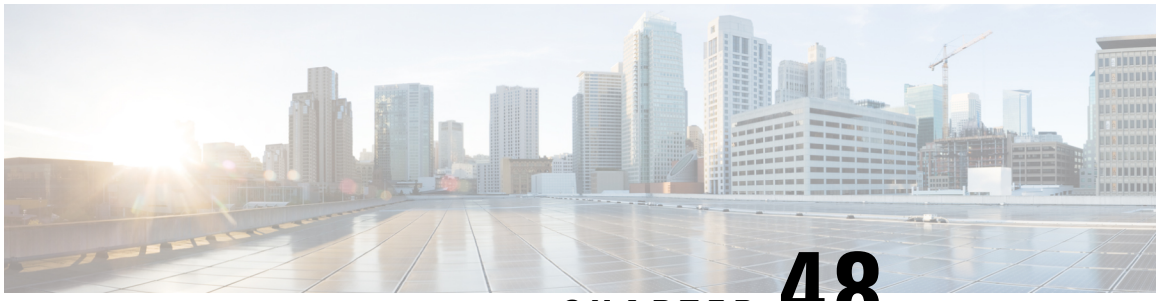
The HTTP client sends requests to both IPv4 and IPv6 HTTP servers, which respond to requests from both IPv4 and IPv6 HTTP clients. URLs with literal IPv6 addresses must be specified in hexadecimal using 16-bit values between colons.

The accept socket call chooses an IPv4 or IPv6 address family. The accept socket is either an IPv4 or IPv6 socket. The listening socket continues to listen for both IPv4 and IPv6 signals that indicate a connection. The IPv6 listening socket is bound to an IPv6 wildcard address.

The underlying TCP/IP stack supports a dual-stack environment. HTTP relies on the TCP/IP stack and the sockets for processing network-layer interactions.

Basic network connectivity (**ping**) must exist between the client and the server hosts before HTTP connections can be made.

For more information, see the "Managing Cisco IOS Applications over IPv6" chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.



CHAPTER 48

IPv6 Client IP Address Learning

Prerequisites for IPv6 Client Address Learning

Before configuring IPv6 client address learning, configure the clients to support IPv6.

Information About IPv6 Client Address Learning

Client Address Learning is configured on device to learn the client's IPv4 and IPv6 address and clients transition state maintained by the device on an association, re-association, de-authentication and timeout.

There are three ways for IPv6 client to acquire IPv6 addresses:

- Stateless Address Auto-Configuration (SLACC)
- Stateful DHCPv6
- Static Configuration

For all of these methods, the IPv6 client always sends neighbor solicitation DAD (Duplicate Address Detection) request to ensure there is no duplicate IP address on the network. The device snoops the client's Neighbor Discovery Protocol (NDP) and DHCPv6 packets to learn about its client IP addresses.

When a duplicate IPv6 address is configured, DAD detects the duplicate address, and advertises it in the Router Advertisement (RA). The duplicate address can be manually removed from the system, so that it is not displayed in the connected address and not advertised in the RA prefix.

SLAAC Address Assignment

The most common method for IPv6 client address assignment is Stateless Address Auto-Configuration (SLAAC). SLAAC provides simple plug-and-play connectivity where clients self-assign an address based on the IPv6 prefix. This process is achieved

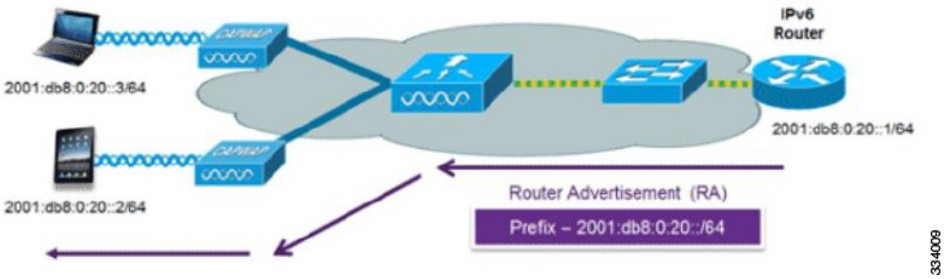
Stateless Address Auto-Configuration (SLAAC) is configured as follows:

- Host sends a router solicitation message.
- Hosts waits for a Router Advertisement message.
- Hosts take the first 64 bits of the IPv6 prefix from the Router Advertisement message and combines it with the 64 bit EUI-64 address (in the case of ethernet, this is created from the MAC Address) to create a global unicast message. The host also uses the source IP address, in the IP header, of the Router Advertisement message, as its default gateway.
- Duplicate Address Detection is performed by IPv6 clients in order to ensure that random addresses that are picked do not collide with other clients.
- The choice of algorithm is up to the client and is often configurable.

The last 64 bits of the IP v6 address can be learned based on the following 2 algorithms:

- EUI-64 which is based on the MAC address of the interface, or
- Private addresses that are randomly generated.

Figure 52: SLAAC Address Assignment

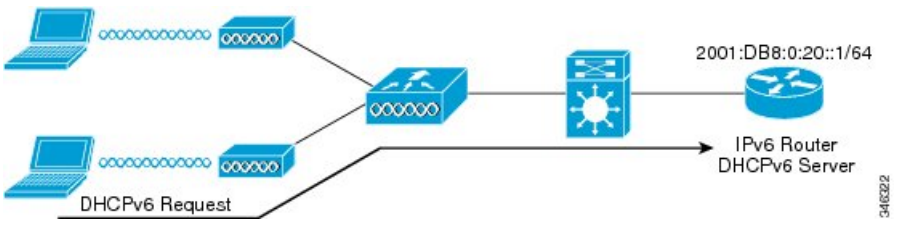


The following Cisco IOS configuration commands from a Cisco-capable IPv6 router are used to enable SLAAC addressing and router advertisements:

```
ipv6 unicast-routing
interface Vlan20
description IPv6-SLAAC
ip address 192.168.20.1 255.255.255.0
ipv6 address FE80:DB8:0:20::1 linklocal
ipv6 address 2001:DB8:0:20::1/64
ipv6 enable
end
```

Stateful DHCPv6 Address Assignment

Figure 53: Stateful DHCPv6 Address Assignment



The use of DHCPv6 is not required for IPv6 client connectivity if SLAAC is already deployed. There are two modes of operation for DHCPv6 called Stateless and Stateful.

The DHCPv6 Stateless mode is used to provide clients with additional network information that is not available in the router advertisement, but not an IPv6 address as this is already provided by SLAAC. This information can include the DNS domain name, DNS server(s), and other DHCP vendor-specific options. This interface configuration is for a Cisco IOS IPv6 router implementing stateless DHCPv6 with SLAAC enabled:

```
ipv6 unicast-routing
ipv6 dhcp pool IPV6_DHCPPOOL
address prefix 2001:db8:5:10::/64
domain-name cisco.com
dns-server 2001:db8:6:6::1
interface Vlan20
description IPv6-DHCP-Stateless
ip address 192.168.20.1 255.255.255.0
ipv6 nd other-config-flag
ipv6 dhcp server IPV6_DHCPPOOL
ipv6 address 2001:DB8:0:20::1/64
end
```

The DHCPv6 Stateful option, also known as managed mode, operates similarly to DHCPv4 in that it assigns unique addresses to each client instead of the client generating the last 64 bits of the address as in SLAAC. This interface configuration is for a Cisco IOS IPv6 router implementing stateful DHCPv6 on a local device:

```
ipv6 unicast-routing
ipv6 dhcp pool IPV6_DHCPPOOL
address prefix 2001:db8:5:10::/64
domain-name cisco.com
dns-server 2001:db8:6:6::1
interface Vlan20
description IPv6-DHCP-Stateful
ip address 192.168.20.1 255.255.255.0
ipv6 address 2001:DB8:0:20::1/64
ipv6 nd prefix 2001:DB8:0:20::/64 no-advertise
ipv6 nd managed-config-flag
ipv6 nd other-config-flag
ipv6 dhcp server IPV6_DHCPPOOL
end
```

This interface configuration is for a Cisco IOS IPv6 router implementing stateful DHCPv6 on an external DHCP server:

```
ipv6 unicast-routing
domain-name cisco.com
dns-server 2001:db8:6:6::1
interface Vlan20
description IPv6-DHCP-Stateful
ip address 192.168.20.1 255.255.255.0
ipv6 address 2001:DB8:0:20::1/64
ipv6 nd prefix 2001:DB8:0:20::/64 no-advertise
ipv6 nd managed-config-flag
ipv6 nd other-config-flag
ipv6 dhcp relay destination 2001:DB8:0:20::2
end
```

Static IP Address Assignment

Statically configured address on a client.

Router Solicitation

A Router Solicitation message is issued by a host to facilitate local routers to transmit Router Advertisement from which it can obtain information about local routing or perform Stateless Auto-configuration. Router Advertisements are transmitted periodically and the host prompts with an immediate Router Advertisement using a Router Solicitation such as - when it boots or following a restart operation.

Router Advertisement

A Router Advertisement message is issued periodically by a router or in response to a Router Solicitation message from a host. The information contained in these messages is used by hosts to perform Stateless Auto-configuration and to modify its routing table.

Neighbor Discovery

IPv6 Neighbor Discovery is a set of messages and processes that determine relationships between neighboring nodes. Neighbor Discovery replaces ARP, ICMP Router Discovery, and ICMP Redirect used in IPv4.

IPv6 Neighbor Discovery inspection analyzes neighbor discovery messages in order to build a trusted binding table database, and IPv6 neighbor discovery packets that do not comply are dropped. The neighbor binding table in the switch tracks each IPv6 address and its associated MAC address. Clients are expired from the table according to Neighbor Binding timers.

Neighbor Discovery Suppression

The IPv6 addresses of clients are cached by the device. When the device receives an NS multicast looking for an IPv6 address, and if the target address is known to the device and belongs to one of its clients, the device will reply with an NA message on behalf of the client. The result of this process generates the equivalent of the Address Resolution Protocol (ARP) table of IPv4 but is more efficient - uses generally fewer messages.

Note The device acts like proxy and respond with NA, only when the **ipv6 nd suppress** command is configured

If the device does not have the IPv6 address of a client, the device will not respond with NA and forward the NS packet . To resolve this, an NS Multicast Forwarding knob is provided. If this knob is enabled, the device gets the NS packet for the IPv6 address that it does not have (cache miss) and forwards it . This packet reaches the intended client and the client replies with NA.

This cache miss scenario occurs rarely, and only very few clients which do not implement complete IPv6 stack may not advertise their IPv6 address during NDP.

RA Guard

IPv6 clients configure IPv6 addresses and populate their router tables based on IPv6 router advertisement (RA) packets. The RA guard feature is similar to the RA guard feature of wired networks. RA guard increases the security of the IPv6 network by dropping the unwanted or rogue RA packets that come from clients. If this feature is not configured, malicious IPv6 clients announce themselves as the router for the network often with high priority, which would take higher precedence over legitimate IPv6 routers.

RA-Guard also examines the incoming RA's and decides whether to switch or block them based solely on information found in the message or in the switch configuration. The information available in the frames received is useful for RA validation:

- Port on which the frame is received
- IPv6 source address
- Prefix list

The following configuration information created on the switch is available to RA-Guard to validate against the information found in the received RA frame:

- Trusted/Untrusted ports for receiving RA-guard messages
- Trusted/Untrusted IPv6 source addresses of RA-sender
- Trusted/Untrusted Prefix list and Prefix ranges
- Router Preference

RA guard is applied on the device. You can configure the device to drop RA messages on the device. All IPv6 RA messages are dropped, which protects other clients and upstream wired network from malicious IPv6 clients.

```
//Create a policy for RA Guard//
ipv6 nd raguard policy raguard-router
trusted-port
device-role router

//Applying the RA Guard Policy on port/interface//
interface gigabitethernet1/1 (Katana)
interface gigabitethernet1/1 (Edison)

ipv6 nd raguard attach-policy raguard-router
```

How to Configure IPv6 Client Address Learning

The following sections provide configuration information about IPv6 client address learning.

Configuring IPv6 Unicast

IPv6 unicasting must always be enabled on the switch. IPv6 unicast routing is disabled.

To configure IPv6 unicast, perform this procedure:

Before you begin

To enable the forwarding of IPv6 unicast datagrams, use the **ipv6 unicast-routing** command in global configuration mode. To disable the forwarding of IPv6 unicast datagrams, use the **no** form of this command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast routing** **Example:** `Device(config)# ipv6 unicast routing`	enable the forwarding of IPv6 unicast datagrams

Configuring RA Guard Policy

Configure RA Guard policy on the device to add IPv6 client addresses and populate the router table based on IPv6 router advertisement packets.

To configuring RA guard policy, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 nd raguard policy raguard-router** **Example:** `Device(config)# ipv6 nd raguard policy raguard-router`	Defines the RA guard policy name and enters RA guard policy configuration mode.
Step 4	**trustedport** **Example:**	(Optional) Specifies that this policy is being applied to trusted ports.

	Command or Action	Purpose
	`Device(config-ra-guard)# trustedport`	
Step 5	**device-role router** **Example:** `Device(config-ra-guard)# device-role router`	Specifies the role of the device attached to the port.
Step 6	**exit** **Example:** `Device(config-ra-guard)# exit`	Exits RA guard policy configuration mode and returns to global configuration mode.

Applying RA Guard Policy

Applying the RA Guard policy on the device will block all the untrusted RA's.

To apply RA guard policy, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet 1/1** **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 nd raguard attach-policy raguard-router** **Example:** `Device(config-if)# ipv6 nd raguard attach-policy raguard-router`	Applies the IPv6 RA Guard feature to a specified interface.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.

Configuring IPv6 Snooping

Note We recommend that you configure SISF-based device tracking configurations instead of IPv6 snooping legacy configuration. For more information, refer to the *Configuring SISF-Based Device Tracking* section in the *Security Configuration Guide*.

IPv6 snooping must always be enabled on the device.

To configuring IPv6 snooping, perform this procedure:

Before you begin

Enable IPv6 on the client machine.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan configuration 1** **Example:** `Device(config)# vlan configuration 1`	Enters VLAN configuration mode.
Step 4	**ipv6 snooping** **Example:** `Device(config-vlan)# ipv6 snooping`	Enables IPv6 snooping on the VLAN.
Step 5	**ipv6 nd suppress** **Example:** `Device(config-vlan-config)# ipv6 nd suppress`	Enables the IPv6 ND suppress on the VLAN.
Step 6	**exit** **Example:** `Device(config-vlan-config)# exit`	Saves the configuration and comes out of the VLAN configuration mode.

Configuring IPv6 ND Suppress Policy

The IPv6 neighbor discovery (ND) multicast suppress feature stops as many ND multicast neighbor solicit (NS) messages as possible by dropping them (and responding to solicitations on behalf of the targets) or converting them into unicast traffic. This feature runs on a layer 2 switch and is used to reduce the amount of control traffic necessary for proper link operations.

When an address is inserted into the binding table, an address resolution request sent to a multicast address is intercepted, and the device either responds on behalf of the address owner or, at layer 2, converts the request into a unicast message and forwards it to its destination.

To configure IPv6 ND suppress policy, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 nd suppress policy** *policy_name* **Example:** `Device(config)# ipv6 nd suppress policy policy1`	Defines the ND suppress policy name and enters ND suppress policy configuration mode.

Configuring IPv6 Snooping on VLAN/PortChannel

Neighbor Discover (ND) suppress can be enabled or disabled on either the VLAN or a switchport.

To configure IPv6 snooping on VLAN/PortChannel, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**vlan config901** **Example:** `Device(config)# vlan config901`	Creates a VLAN and enter the VLAN configuration mode
Step 4	**ipv6 nd suppress** **Example:** `Device(config-vlan)# ipv6 nd suppress`	Applies the IPv6 nd suppress on VLAN.
Step 5	**end** **Example:** `Device(config-vlan)# end`	Exits vlan configuration mode and enters the global configuration mode.
Step 6	**interface gigabitethernet 1/1** **Example:** `Device(config)# interface gigabitethernet 1/1`	Creates a gigabitethernet interface.
Step 7	**ipv6 nd suppress** **Example:** `Device(config-vlan)# ipv6 nd suppress`	Applies the IPv6 nd suppress on the interface.
Step 8	**end** **Example:** `Device(config-vlan)# end`	Exits vlan configuration mode and enters the global configuration mode.

Configuring IPv6 on Switch Interface

Follow the procedure given below to configure IPv6 on an interface:

Before you begin

Enable IPv6 on the client and IPv6 support on the wired infrastructure.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface vlan 1** **Example:** `Device(config)# interface vlan 1`	Creates a interface and enters interface configuration mode.
Step 4	**ip address fe80::1 link-local** **Example:** `Device(config-if)# ip address 198.51.100.1 255.255.255.0` `Device(config-if)# ipv6 address fe80::1 link-local` `Device(config-if)# ipv6 address 2001:DB8:0:1:FFFF:1234::5/64` `Device(config-if)# ipv6 address 2001:DB8:0:0:E000::F/64`	Configures IPv6 address on the interface using the link-local option.
Step 5	**ipv6 enable** **Example:** `Device(config)# ipv6 enable`	(Optional) Enables IPv6 on the interface.
Step 6	**end** **Example:** `Device(config)# end`	Exits from the interface mode.

Configuring DHCP Pool on Switch Interface

Follow the procedure given below to configure DHCP Pool on an interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 dhcp pool Vlan21** **Example:** `Device(config)# ipv6 dhcp pool vlan1`	Enters the configuration mode and configures the IPv6 DHCP pool on the Vlan.
Step 4	**address prefix 2001:DB8:0:1:FFFF:1234::/64 lifetime 300 10** **Example:** `Device(config-dhcpv6)# address prefix 2001:DB8:0:1:FFFF:1234::/64 lifetime 300 10`	Enters the configuration-dhcp mode and configures the address pool and its lifetime on a Vlan.
Step 5	**dns-server 2001:100:0:1::1** **Example:** `Device(config-dhcpv6)# dns-server 2001:20:21::1`	Configures the DNS servers for the DHCP pool.
Step 6	**domain-name example.com** **Example:** `Device(config-dhcpv6)# domain-name example.com`	Configures the domain name to complete unqualified host names.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

Configuring Stateless Auto Address Configuration Without DHCP

Follow the procedure given below to configure stateless auto address configuration without DHCP:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface vlan 1** **Example:** `Device(config)# interface vlan 1`	Creates a interface and enters interface configuration mode.
Step 4	**ip address fe80::1 link-local** **Example:** `Device(config-if)# ip address 198.51.100.1 255.255.255.0` `Device(config-if)# ipv6 address fe80::1 link-local` `Device(config-if)# ipv6 address 2001:DB8:0:1:FFFF:1234::5/64` `Device(config-if)# ipv6 address 2001:DB8:0:0:E000::F/64`	Configures IPv6 address on the interface using the link-local option.
Step 5	**ipv6 enable** **Example:** `Device(config)# ipv6 enable`	(Optional) Enables IPv6 on the interface.
Step 6	**no ipv6 nd managed-config-flag** **Example:** `Device(config)# interface vlan 1` `Device(config-if)# no ipv6 nd managed-config-flag`	Ensures the attached hosts do not use stateful autoconfiguration to obtain addresses.
Step 7	**no ipv6 nd other-config-flag** **Example:** `Device(config-if)# no ipv6 nd other-config-flag`	Ensures the attached hosts do not use stateful autoconfiguration to obtain non-address options from DHCP (domain etc).
Step 8	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

Configuring Stateless Auto Address Configuration With DHCP

Follow the procedure given below to configure stateless auto address configuration with DHCP:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface vlan 1** **Example:** `Device(config)# interface vlan 1`	Creates a interface and enters interface configuration mode.
Step 4	**ip address fe80::1 link-local** **Example:** `Device(config-if)# ip address 198.51.100.1 255.255.255.0` `Device(config-if)# ipv6 address fe80::1 link-local` `Device(config-if)# ipv6 address 2001:DB8:0:1:FFFF:1234::5/64` `Device(config-if)# ipv6 address 2001:DB8:0:0:E000::F/64`	Configures IPv6 address on the interface using the link-local option.
Step 5	**ipv6 enable** **Example:** `Device(config)# ipv6 enable`	(Optional) Enables IPv6 on the interface.
Step 6	**no ipv6 nd managed-config-flag** **Example:** `Device(config)# interface vlan 1` `Device(config-if)# no ipv6 nd managed-config-flag`	Ensures the attached hosts do not use stateful autoconfiguration to obtain addresses.
Step 7	**ipv6 nd other-config-flag** **Example:** `Device(config-if)# no ipv6 nd other-config-flag`	Ensures the attached hosts do not use stateful autoconfiguration to obtain non-address options from DHCP (domain etc).
Step 8	**end** **Example:** `Device(config)# end`	Exits from the interface mode.

Configuring Stateful DHCP Locally

This interface configuration is for a Cisco IOS IPv6 router implementing stateful DHCPv6 on a local device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Configures IPv6 for unicasting.
Step 4	**ipv6 dhcp pool IPv6_DHCPPOOL** **Example:** `Device(config)# ipv6 dhcp pool IPv6_DHCPPOOL`	Enters the configuration mode and configures the IPv6 DHCP pool on the VLAN.
Step 5	**address prefix 2001:DB8:0:1:FFFF:1234::/64** **Example:** `Device(config-dhcpv6)# address prefix 2001:DB8:0:1:FFFF:1234::/64`	Specifies the address range to provide in the pool.
Step 6	**dns-server 2001:100:0:1::1** **Example:** `Device(config-dhcpv6)# dns-server 2001:100:0:1::1`	Provides the DNS server option to DHCP clients.
Step 7	**domain-name example.com** **Example:** `Device(config-dhcpv6)# domain-name example.com`	Provides the domain name option to DHCP clients.
Step 8	**exit** **Example:** `Device(config-dhcpv6)# exit`	Returns to the previous mode.
Step 9	**interface vlan1** **Example:** `Device(config)# interface vlan 1`	Enters the interface mode to configure the stateful DHCP.

	Command or Action	Purpose
Step 10	**description IPv6-DHCP-Stateful** **Example:** `Device(config-if)# description IPv6-DHCP-Stateful`	Enter description for the stateful IPv6 DHCP.
Step 11	**ipv6 address 2001:DB8:0:20::1/64** **Example:** `Device(config-if)# ipv6 address 2001:DB8:0:20::1/64`	Enters the IPv6 address for the stateful IPv6 DHCP.
Step 12	**ip address 192.168.20.1 255.255.255.0** **Example:** `Device(config-if)# ip address 192.168.20.1 255.255.255.0`	Enters the IPv6 address for the stateful IPv6 DHCP.
Step 13	**ipv6 nd prefix 2001:db8::/64 no-advertise** **Example:** `Device(config-if)# ipv6 nd prefix 2001:db8::/64 no-advertise`	Configures the IPv6 routing prefix advertisement that must not be advertised.
Step 14	**ipv6 nd managed-config-flag** **Example:** `Device(config-if)# ipv6 nd managed-config-flag`	Configures IPv6 interfaces neighbor discovery to allow the hosts to uses DHCP for address configuration.
Step 15	**ipv6 nd other-config-flag** **Example:** `Device(config-if)# ipv6 nd other-config-flag`	Configures IPv6 interfaces neighbor discovery to allow the hosts to uses DHCP for non-address configuration.
Step 16	**ipv6 dhcp server IPv6_DHCPPOOL** **Example:** `Device(config-if)# ipv6 dhcp server IPv6_DHCPPOOL`	Configures the DHCP server on the interface.

Configuring Stateful DHCP Externally

This interface configuration is for a Cisco IOS IPv6 router implementing stateful DHCPv6 on an external DHCP server.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Configures the IPv6 for unicasting.
Step 4	**dns-server** `2001:100:0:1::1` **Example:** `Device(config-dhcpv6)# dns-server 2001:100:0:1::1`	Provides the DNS server option to DHCP clients.
Step 5	**domain-name** `example.com` **Example:** `Device(config-dhcpv6)# domain-name example.com`	Provides the domain name option to DHCP clients.
Step 6	**exit** **Example:** `Device(config-dhcpv6)# exit`	Returns to the previous mode.
Step 7	**interface** `vlan 1` **Example:** `Device(config)# interface vlan 1`	Enters the interface mode to configure the stateful DHCP.
Step 8	**description IPv6-DHCP-Stateful** **Example:** `Device(config-if)# description IPv6-DHCP-Stateful`	Enter description for the stateful IPv6 DHCP.
Step 9	**ipv6 address** `2001:DB8:0:20::1/64` **Example:** `Device(config-if)# ipv6 address 2001:DB8:0:20::1/64`	Enters the IPv6 address for the stateful IPv6 DHCP.
Step 10	**ip address** `192.168.20.1 255.255.255.0` **Example:**	Enters the IPv6 address for the stateful IPv6 DHCP.

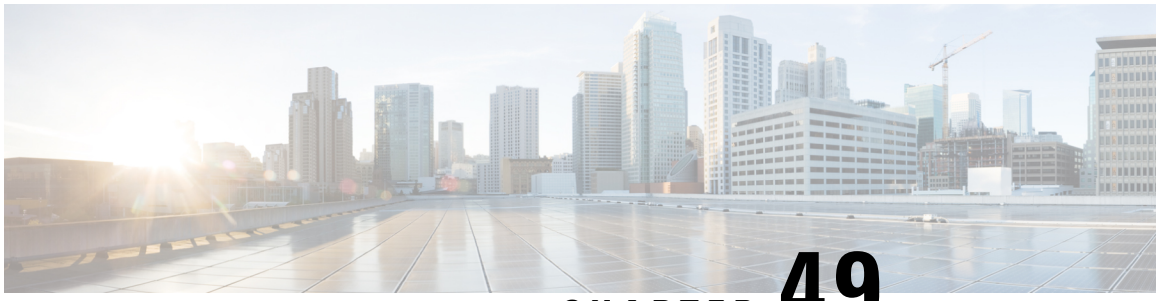
	Command or Action	Purpose
	`Device(config-if)# ip address 192.168.20.1 255.255.255.0`	
Step 11	**ipv6 nd prefix 2001:db8::/64 no-advertise** **Example:** `Device(config-if)# ipv6 nd prefix 2001:db8::/64 no-advertise`	Configures the IPv6 routing prefix advertisement that must not be advertised.
Step 12	**ipv6 nd managed-config-flag** **Example:** `Device(config-if)# ipv6 nd managed-config-flag`	Configures IPv6 interfaces neighbor discovery to allow the hosts to uses DHCP for address configuration.
Step 13	**ipv6 nd other-config-flag** **Example:** `Device(config-if)# ipv6 nd other-config-flag`	Configures IPv6 interfaces neighbor discovery to allow the hosts to uses DHCP for non-address configuration.
Step 14	**ipv6 dhcp relaydestination 2001:DB8:0:20::2** **Example:** `Device(config-if)# ipv6 dhcp relay destination 2001:DB8:0:20::2`	Configures the DHCP server on the interface.

Verifying IPv6 Address Learning Configuration

This example displays the output of the **show ipv6 dhcp pool** command. This command displays the IPv6 service configuration on the device. The vlan 21 configured pool detail displays 6 clients that are currently using addresses from the pool.

Procedure

	Command or Action	Purpose
Step 1	**show ipv6 dhcp pool** **Example:** `Device show ipv6 dhcp pool` `DHCPv6 pool: vlan21` `Address allocation prefix: 2001:DB8:0:1:FFFF:1234::/64 valid 86400 preferred 86400 (6 in use, 0 conflicts)` `DNS server: 2001:100:0:1::1` `Domain name: example.com` `Active clients: 6`	Displays the IPv6 service configuration on the device.



CHAPTER 49

Configuring DHCP

This section provides information about configuring DHCP.

Prerequisites for Configuring DHCP

The following prerequisites apply to DHCP Snooping and Option 82:

- You must globally enable DHCP snooping on the switch.
- Before globally enabling DHCP snooping on the switch, make sure that the devices acting as the DHCP server and the DHCP relay agent are configured and enabled.
- If you want the switch to respond to DHCP requests, it must be configured as a DHCP server.
- Before configuring the DHCP snooping information option on your switch, be sure to configure the device that is acting as the DHCP server. You must specify the IP addresses that the DHCP server can assign or exclude, or you must configure DHCP options for these devices.
- For DHCP snooping to function properly, all DHCP servers must be connected to the switch through trusted interfaces, as untrusted DHCP messages will be forwarded only to trusted interfaces. In a service-provider network, a trusted interface is connected to a port on a device in the same network.
- You must configure the switch to use the Cisco IOS DHCP server binding database to use it for DHCP snooping.
- To use the DHCP snooping option of accepting packets on untrusted inputs, the switch must be an aggregation switch that receives packets with option-82 information from an edge switch.
- The following prerequisites apply to DHCP snooping binding database configuration:
  - You must configure a destination on the DHCP snooping binding database to use the switch for DHCP snooping.
  - Because both NVRAM and the flash memory have limited storage capacity, we recommend that you store the binding file on a TFTP server.

  - For network-based URLs (such as TFTP and FTP), you must create an empty file at the configured URL before the switch can write bindings to the binding file at that URL. See the documentation for your TFTP server to determine whether you must first create an empty file on the server; some TFTP servers cannot be configured this way.
  - To ensure that the lease time in the database is accurate, we recommend that you enable and configure Network Time Protocol (NTP).
  - If NTP is configured, the switch writes binding changes to the binding file only when the switch system clock is synchronized with NTP.

- Before configuring the DHCP relay agent on your switch, make sure to configure the device that is acting as the DHCP server. You must specify the IP addresses that the DHCP server can assign or exclude, configure DHCP options for devices, or set up the DHCP database agent.
- If you want the switch to relay DHCP packets, the IP address of the DHCP server must be configured on the switch virtual interface (SVI) of the DHCP client.
- If a switch port is connected to a DHCP server, configure a port as trusted by entering the **ip dhcp snooping trust** interface configuration command.
- If a switch port is connected to a DHCP client, configure a port as untrusted by entering the **no ip dhcp snooping trust** interface configuration command.

Restrictions for Configuring DHCP

We recommend that you do not use transmit (TX) Remote or Encapsulated Remote Switched Port Analyzer (RSPAN or ERSPAN) on VLAN ports which support DHCP Snooping or DHCP Relay Agent. If TX RSPAN or ERSPAN is required, avoid using VLAN ports that are in the forwarding path for DHCP packets.

Information About DHCP

DHCP Server

The DHCP server assigns IP addresses from specified address pools on a switch or router to DHCP clients and manages them. If the DHCP server cannot give the DHCP client the requested configuration parameters from its database, it forwards the request to one or more secondary DHCP servers defined by the network administrator. The switch can act as a DHCP server. If the DHCP server provides the client with the requested configuration, it will not forward the message to the other server.

DHCP Relay Agent

A DHCP relay agent is a Layer 3 device that forwards DHCP packets between clients and servers. Relay agents forward requests and replies between clients and servers when they are not on the same physical subnet. Relay agent forwarding is different from the normal Layer 2 forwarding, in which IP datagrams are switched transparently between networks. Relay agents receive DHCP messages and generate new DHCP messages to send on output interfaces.

DHCP Snooping

DHCP snooping is a DHCP security feature that provides network security by filtering untrusted DHCP messages and by building and maintaining a DHCP snooping binding database, also referred to as a DHCP snooping binding table.

DHCP snooping acts like a firewall between untrusted hosts and DHCP servers. You use DHCP snooping to differentiate between untrusted interfaces connected to the end user and trusted interfaces connected to the DHCP server or another switch.

Note For DHCP snooping to function properly, all DHCP servers must be connected to the switch through trusted interfaces, as untrusted DHCP messages will be forwarded only to trusted interfaces.

An untrusted DHCP message is a message that is received through an untrusted interface. By default, the switch considers all interfaces untrusted. So, the switch must be configured to trust some interfaces to use DHCP Snooping. When you use DHCP snooping in a service-provider environment, an untrusted message is sent from a device that is not in the service-provider network, such as a customer's switch. Messages from unknown devices are untrusted because they can be sources of traffic attacks.

The DHCP snooping binding database has the MAC address, the IP address, the lease time, the binding type, the VLAN number, and the interface information that corresponds to the local untrusted interfaces of a switch. It does not have information regarding hosts interconnected with a trusted interface.

In a service-provider network, an example of an interface you might configure as trusted is one connected to a port on a device in the same network. An example of an untrusted interface is one that is connected to an untrusted interface in the network or to an interface on a device that is not in the network.

When a switch receives a packet on an untrusted interface and the interface belongs to a VLAN in which DHCP snooping is enabled, the switch compares the source MAC address and the DHCP client hardware address. If the addresses match (the default), the switch forwards the packet. If the addresses do not match, the switch drops the packet.

The switch drops a DHCP packet when one of these situations occurs:

- A packet from a DHCP server, such as a DHCPOFFER, DHCPACK, DHCPNAK, or DHCPLEASEQUERY packet, is received from outside the network or firewall.
- A packet is received on an untrusted interface, and the source MAC address and the DHCP client hardware address do not match.
- The switch receives a DHCPRELEASE or DHCPDECLINE broadcast message that has a MAC address in the DHCP snooping binding database, but the interface information in the binding database does not match the interface on which the message was received.
- A DHCP relay agent forwards a DHCP packet that includes a relay-agent IP address that is not 0.0.0.0, or the relay agent forwards a packet that includes option-82 information to an untrusted port.
- The maximum snooping queue size of 1000 is exceeded when DHCP snooping is enabled.

If the switch is an aggregation switch supporting DHCP snooping and is connected to an edge switch that is inserting DHCP option-82 information, the switch drops packets with option-82 information when packets are received on an untrusted interface. If DHCP snooping is enabled and packets are received on a trusted port, the aggregation switch does not learn the DHCP snooping bindings for connected devices and cannot build a complete DHCP snooping binding database.

When an aggregation switch can be connected to an edge switch through an untrusted interface and you enter the **ip dhcp snooping information option allow-untrusted** global configuration command, the aggregation switch accepts packets with option-82 information from the edge switch. The aggregation switch learns the bindings for hosts connected through an untrusted switch interface. The DHCP security features, such as dynamic ARP inspection or IP source guard, can still be enabled on the aggregation switch while the switch receives packets with option-82 information on untrusted input interfaces to which hosts are connected. The port on the edge switch that connects to the aggregation switch must be configured as a trusted interface.

DHCP Snooping and Local SPAN can be configured on the same VLAN for non-SDA deployments.

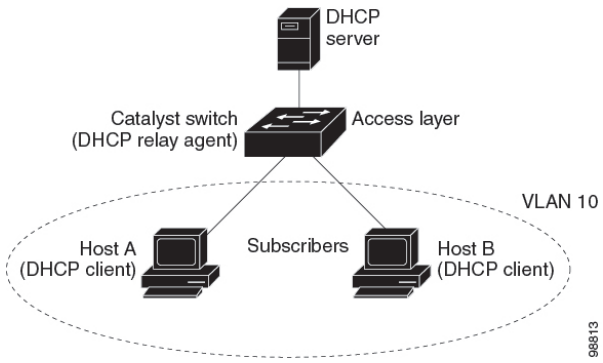
Option-82 Data Insertion

In residential, metropolitan Ethernet-access environments, DHCP can centrally manage the IP address assignments for a large number of subscribers. When the DHCP option-82 feature is enabled on the switch, a subscriber device is identified by the switch port through which it connects to the network (in addition to its MAC address). Multiple hosts on the subscriber LAN can be connected to the same port on the access switch and are uniquely identified.

Note The DHCP option-82 feature is supported only when DHCP snooping is globally enabled on the VLANs to which subscriber devices using option-82 are assigned.

The following illustration shows a metropolitan Ethernet network in which a centralized DHCP server assigns IP addresses to subscribers connected to the switch at the access layer. Because the DHCP clients and their associated DHCP server do not reside on the same IP network or subnet, a DHCP relay agent (the switch) is configured with a helper address to enable broadcast forwarding and to transfer DHCP messages between the clients and the server.

Figure 54: DHCP Relay Agent in a Metropolitan Ethernet Network



When you enable the DHCP snooping information option 82 on the switch, the following sequence of events occurs:

- The host (DHCP client) generates a DHCP request and broadcasts it on the network.
- When the switch receives the DHCP request, it adds the option-82 information in the packet. By default, the remote-ID suboption is the switch MAC address, and the circuit-ID suboption is the port identifier, **vlan-mod-port**, from which the packet is received. You can configure the remote ID and circuit ID.
- If the IP address of the relay agent is configured, the switch adds this IP address in the DHCP packet.

- The switch forwards the DHCP request that includes the option-82 field to the DHCP server.
- The DHCP server receives the packet. If the server is option-82-capable, it can use the remote ID, the circuit ID, or both to assign IP addresses and implement policies, such as restricting the number of IP addresses that can be assigned to a single remote ID or circuit ID. Then the DHCP server echoes the option-82 field in the DHCP reply.
- The DHCP server unicasts the reply to the switch if the request was relayed to the server by the switch. The switch verifies that it originally inserted the option-82 data by inspecting the remote ID and possibly the circuit ID fields. The switch removes the option-82 field and forwards the packet to the switch port that connects to the DHCP client that sent the DHCP request.

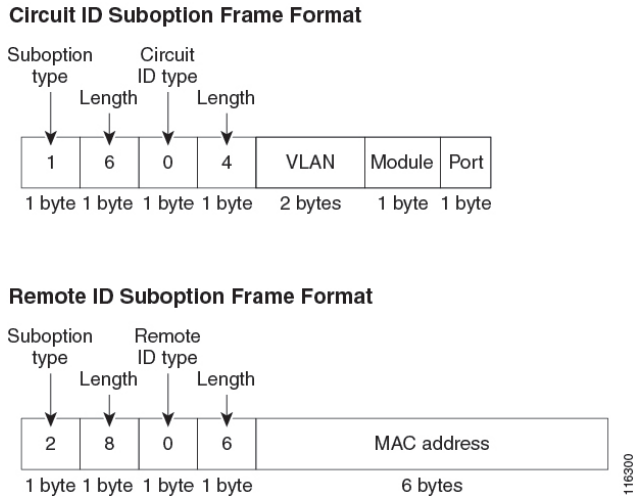
In the default suboption configuration, when the described sequence of events occurs, the values in these fields do not change (see the illustration,*Suboption Packet Formats*):

- Circuit-ID suboption fields
  - Suboption type
  - Length of the suboption type
  - Circuit-ID type
  - Length of the circuit-ID type
- Remote-ID suboption fields
  - Suboption type
  - Length of the suboption type
  - Remote-ID type
  - Length of the remote-ID type

In the port field of the circuit ID suboption, the port numbers start at 3. For example, on a switch with 24 10/100/1000 ports and four small form-factor pluggable (SFP) module slots, port 3 is the Gigabit Ethernet 1/1 port, port 4 is the Gigabit Ethernet 1/2 port, and so forth. Port 27 is the SFP module slot Gigabit Ethernet1/3, and so forth.

The illustration, *Suboption Packet Formats*. shows the packet formats for the remote-ID suboption and the circuit-ID suboption when the default suboption configuration is used. For the circuit-ID suboption, the module number corresponds to the switch number . The switch uses the packet formats when you globally enable DHCP snooping and enter the ip dhcp snooping information option global configuration command.

Figure 55: Suboption Packet Formats

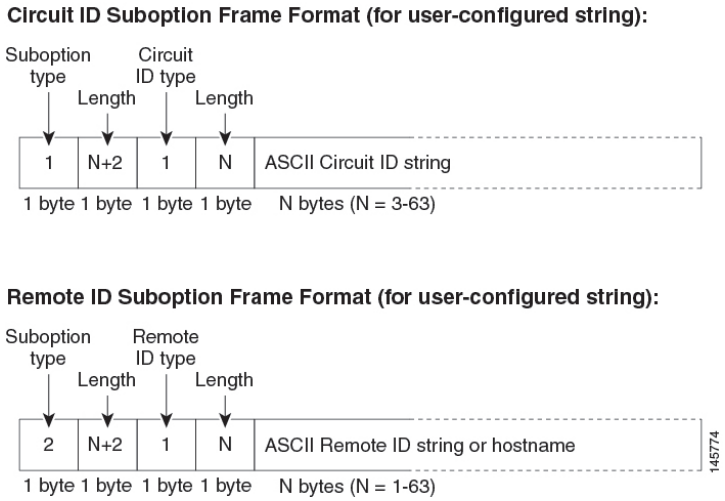


The illustration, *User-Configured Suboption Packet Formats*, shows the packet formats for user-configured remote-ID and circuit-ID suboptions The switch uses these packet formats when DHCP snooping is globally enabled and when the **ip dhcp snooping information option format remote-id** global configuration command and the**ip dhcp snooping vlan information option format-type circuit-id string** interface configuration command are entered.

The values for these fields in the packets change from the default values when you configure the remote-ID and circuit-ID suboptions:

- Circuit-ID suboption fields
  - The circuit-ID type is 1.
  - The length values are variable, depending on the length of the string that you configure.
- Remote-ID suboption fields
  - The remote-ID type is 1.
  - The length values are variable, depending on the length of the string that you configure.

Figure 56: User-Configured Suboption Packet Formats



Cisco IOS DHCP Server Database

During the DHCP-based autoconfiguration process, the designated DHCP server uses the Cisco IOS DHCP server database. It has IP addresses, address bindings, and configuration parameters, such as the boot file.

An address binding is a mapping between an IP address and a MAC address of a host in the Cisco IOS DHCP server database. You can manually assign the client IP address, or the DHCP server can allocate an IP address from a DHCP address pool.

DHCP Snooping Binding Database

When DHCP snooping is enabled, the switch uses the DHCP snooping binding database to store information about untrusted interfaces. The database can have up to 64,000 bindings.

Each database entry (binding) has an IP address, an associated MAC address, the lease time (in hexadecimal format), the interface to which the binding applies, and the VLAN to which the interface belongs. The database agent stores the bindings in a file at a configured location. At the end of each entry is a checksum that accounts for all the bytes from the start of the file through all the bytes associated with the entry. Each entry is 77 bytes, followed by a space, the checksum value, and the EOL symbol.

To keep the bindings when the switch reloads, you must use the DHCP snooping database agent. If the agent is disabled, dynamic ARP inspection or IP source guard is enabled, and the DHCP snooping binding database has dynamic bindings, the switch loses its connectivity. If the agent is disabled and only DHCP snooping is enabled, the switch does not lose its connectivity, but DHCP snooping might not prevent DHCP spoofing attacks.

When reloading, the switch reads the binding file to build the DHCP snooping binding database. The switch updates the file when the database changes.

When a switch learns of new bindings or when it loses bindings, the switch immediately updates the entries in the database. The switch also updates the entries in the binding file. The frequency at which the file is updated is based on a configurable delay, and the updates are batched. If the file is not updated in a specified time (set by the write-delay and abort-timeout values), the update stops.

This is the format of the file with bindings:

```
<initial-checksum>
TYPE DHCP-SNOOPING
VERSION 1
BEGIN
<entry-1> <checksum-1>
<entry-2> <checksum-1-2>
...
...
<entry-n> <checksum-1-2-..-n>
END
```

Each entry in the file is tagged with a checksum value that the switch uses to verify the entries when it reads the file. The initial-checksum entry on the first line distinguishes entries associated with the latest file update from entries associated with a previous file update.

This is an example of a binding file:

```
3ebe1518
TYPE DHCP-SNOOPING
VERSION 1
BEGIN
10.1.1.1 512 001.0001.0005 3EBE2881 Gi1/1                                   e5e1e733
10.1.1.1 512 001.0001.0002 3EBE2881 Gi1/1                                   4b3486ec
10.1.1.1 1536 001.0001.0004 3EBE2881 Gi1/1                                  f0e02872
10.1.1.1 1024 001.0001.0003 3EBE2881 Gi1/1                                  ac41adf9
10.1.1.1 1 001.0001.0001 3EBE2881 Gi1/1                                     34b3273e
END
```

When the switch starts and the calculated checksum value equals the stored checksum value, the switch reads entries from the binding file and adds the bindings to its DHCP snooping binding database. The switch ignores an entry when one of these situations occurs:

- The switch reads the entry and the calculated checksum value does not equal the stored checksum value. The entry and the ones following it are ignored.
- An entry has an expired lease time (the switch might not remove a binding entry when the lease time expires).
- The interface in the entry no longer exists on the system.
- The interface is a routed interface or a DHCP snooping-trusted interface.

Default DHCP Snooping Configuration

Table 56: Default DHCP Configuration

Feature	Default Setting
DHCP server	Enabled in Cisco IOS software, requires configuration[3]
DHCP relay agent	Enabled[4]
DHCP packet forwarding address	None configured
Checking the relay agent information	Enabled (invalid messages are dropped)

Feature	Default Setting
DHCP relay agent forwarding policy	Replace the existing relay agent information
DHCP snooping enabled globally	Disabled
DHCP snooping information option	Enabled
DHCP snooping option to accept packets on untrusted input interfaces[5]	Disabled
DHCP snooping limit rate	None configured
DHCP snooping trust	Untrusted
DHCP snooping VLAN	Disabled
DHCP snooping MAC address verification	Enabled
Cisco IOS DHCP server binding database	Enabled in Cisco IOS software, requires configuration. **Note** The switch gets network addresses and configuration parameters only from a device configured as a DHCP server.
DHCP snooping binding database agent	Enabled in Cisco IOS software, requires configuration. This feature is operational only when a destination is configured.

[3] The switch responds to DHCP requests only if it is configured as a DHCP server.
[4] The switch relays DHCP packets only if the IP address of the DHCP server is configured on the SVI of the DHCP client.
[5] Use this feature when the switch is an aggregation switch that receives packets with option-82 information from an edge switch.

DHCP Snooping Configuration Guidelines

- If a switch port is connected to a DHCP server, configure a port as trusted by entering the **ip dhcp snooping trust** interface configuration command.
- If a switch port is connected to a DHCP client, configure a port as untrusted by entering the **no ip dhcp snooping trust** interface configuration command.
- You can display DHCP snooping statistics by entering the **show ip dhcp snooping statistics** user EXEC command, and you can clear the snooping statistics counters by entering the **clear ip dhcp snooping statistics** privileged EXEC command.

DHCP Server Port-Based Address Allocation

DHCP server port-based address allocation is a feature that enables DHCP to maintain the same IP address on an Ethernet switch port regardless of the attached device client identifier or client hardware address.

When Ethernet switches are deployed in the network, they offer connectivity to the directly connected devices. In some environments, such as on a factory floor, if a device fails, the replacement device must be working immediately in the existing network. With the current DHCP implementation, there is no guarantee that DHCP would offer the same IP address to the replacement device. Control, monitoring, and other software expect a stable IP address associated with each device. If a device is replaced, the address assignment should remain stable even though the DHCP client has changed.

When configured, the DHCP server port-based address allocation feature ensures that the same IP address is always offered to the same connected port even as the client identifier or client hardware address changes in the DHCP messages received on that port. The DHCP protocol recognizes DHCP clients by the client identifier option in the DHCP packet. Clients that do not include the client identifier option are identified by the client hardware address. When you configure this feature, the port name of the interface overrides the client identifier or hardware address and the actual point of connection, the switch port, becomes the client identifier.

In all cases, by connecting the Ethernet cable to the same port, the same IP address is allocated through DHCP to the attached device.

The DHCP server port-based address allocation feature is only supported on a Cisco IOS DHCP server and not a third-party server.

Default Port-Based Address Allocation Configuration

By default, DHCP server port-based address allocation is disabled.

Port-Based Address Allocation Configuration Guidelines

- By default, DHCP server port-based address allocation is disabled.
- To restrict assignments from the DHCP pool to preconfigured reservations (unreserved addresses are not offered to the client and other clients are not served by the pool), you can enter the **reserved-only** DHCP pool configuration command.

How to Configure DHCP

Configuring the DHCP Server

The switch can act as a DHCP server. If DHCP server for DHCP clients with management ports are used, both DHCP pool and the corresponding interface must be configured using the Management VRF.

Configuring the DHCP Relay Agent

Follow these steps to enable the DHCP relay agent on the switch:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service dhcp** **Example:** `Device(config)# service dhcp`	Enables the DHCP server and relay agent on your switch. By default, this feature is enabled.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

What to do next

- Checking (validating) the relay agent information
- Configuring the relay agent forwarding policy

Specifying the Packet Forwarding Address

If the DHCP server and the DHCP clients are on different networks or subnets, you must configure the switch with the **ip helper-address** *address* interface configuration command. The general rule is to configure the command on the Layer 3 interface closest to the client. The address used in the **ip helper-address** command can be a specific DHCP server IP address, or it can be the network address if other DHCP servers are on the destination network segment. Using the network address enables any DHCP server to respond to requests.

Perform these steps to specify the packet forwarding address:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface vlan** *vlan-id* **Example:** `Device(config)# interface vlan 1`	Creates a switch virtual interface by entering a VLAN ID, and enters interface configuration mode.
Step 4	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 192.108.1.27 255.255.255.0`	Configures the interface with an IP address and an IP subnet.
Step 5	**ip helper-address** *address* **Example:** `Device(config-if)# ip helper-address 172.16.1.2`	Specifies the DHCP packet forwarding address. • The helper address can be a specific DHCP server address, or it can be the network address if other DHCP servers are on the destination network segment. Using the network address enables other servers to respond to DHCP requests. • If you have multiple servers, you can configure one helper address for each server.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 7	Use one of the following: • **interface range** *port-range* • **interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/2`	Configures multiple physical ports that are connected to the DHCP clients, and enters interface range configuration mode. or Configures a single physical port that is connected to the DHCP client, and enter interface configuration mode.
Step 8	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Defines the VLAN membership mode for the port.
Step 9	**switchport access vlan** *vlan-id* **Example:**	Assigns the ports to the same VLAN as configured in Step 2.

	Command or Action	Purpose
	`Device(config-if)# switchport access vlan 1`	
Step 10	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring DHCP for IPv6 Address Assignment

Default DHCPv6 Address Assignment Configuration

By default, no DHCPv6 features are configured on the switch.

DHCPv6 Address Assignment Configuration Guidelines

The following prerequisites apply when configuring DHCPv6 address assignment:

- In the following procedures, the specified interface must be one of these Layer 3 interfaces:
  - If the IPv6 address is not explicitly configured, enable IPv6 routing by using the **ipv6 enable** command.
  - DHCPv6 routing must be enabled on a Layer 3 interface.
  - SVI: A VLAN interface created by using the **interface vlan** *vlan_id* command.
  - EtherChannel port channel in Layer 3 mode: a port-channel logical interface created by using the **interface port-channel port-channel-number** command.
- The device can act as a DHCPv6 client, server, or relay agent. The DHCPv6 client, server, and relay function are mutually exclusive on an interface.
- A DHCPv6 address will contain interface identifiers that are not part of the reserved interface identifiers range specified in RFC5453.

Enabling DHCPv6 Server Function (CLI)

Use the **no** form of the DHCP pool configuration mode commands to change the DHCPv6 pool characteristics. To disable the DHCPv6 server function on an interface, use the **no ipv6 dhcp server** interface configuration command.

To enable the DHCPv6 server function on an interface, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Device(config)# ipv6 dhcp pool 7`	Enters DHCP pool configuration mode, and define the name for the IPv6 DHCP pool. The pool name can be a symbolic string (such as Engineering) or an integer (such as 0).
Step 4	**address prefix** *IPv6-prefix* {**lifetime**} {*t1 t1* | **infinite**} **Example:** `Device(config-dhcpv6)# address prefix 2001:1000::0/64 lifetime 3600`	(Optional) Specifies an address prefix for address assignment. This address must be in hexadecimal, using 16-bit values between colons. **lifetime** *t1 t1*—Specifies a time interval (in seconds) that an IPv6 address prefix remains in the valid state. The range is 5 to 4294967295 seconds. Specify **infinite** for no time interval.
Step 5	**link-address** *IPv6-prefix* **Example:** `Device(config-dhcpv6)# link-address 2001:1002::0/64`	(Optional) Specifies a link-address IPv6 prefix. When an address on the incoming interface or a link-address in the packet matches the specified IPv6 prefix, the server uses the configuration information pool. This address must be in hexadecimal, using 16-bit values between colons.
Step 6	**vendor-specific** *vendor-id* **Example:** `Device(config-dhcpv6)# vendor-specific 9`	(Optional) Enters vendor-specific configuration mode and specifies a vendor-specific identification number. This number is the vendor IANA Private Enterprise Number. The range is 1 to 4294967295.
Step 7	**suboption** *number* {**address** *IPv6-address* | **ascii** *ASCII-string* | **hex** *hex-string*} **Example:** `Device(config-dhcpv6-vs)# suboption 1 address 1000:235D::`	(Optional) Enters a vendor-specific suboption number. The range is 1 to 65535. Enter an IPv6 address, ASCII text, or a hex string as defined by the suboption parameters.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Device(config-dhcpv6-vs)# exit`	Returns to DHCP pool configuration mode.
Step 9	**exit** **Example:** `Device(config-dhcpv6)# exit`	Returns to global configuration mode.
Step 10	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the interface to configure.
Step 11	**ipv6 dhcp server** [*poolname* | **automatic**] [**rapid-commit**] [**preference** *value*] [**allow-hint**] **Example:** `Device(config-if)# ipv6 dhcp server automatic`	Enables DHCPv6 server function on an interface. • *poolname*—(Optional) User-defined name for the IPv6 DHCP pool. The pool name can be a symbolic string (such as Engineering) or an integer (such as 0). • **automatic**—(Optional) Enables the system to automatically determine which pool to use when allocating addresses for a client. • **rapid-commit**—(Optional) Allows two-message exchange method. • **preference** *value*—(Optional) Configures the preference value carried in the preference option in the advertise message sent by the server. The range is from 0 to 255. The preference value default is 0. • **allow-hint**—(Optional) Specifies whether the server should consider client suggestions in the SOLICIT message. By default, the server ignores client hints.
Step 12	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
Step 13	Do one of the following: • **show ipv6 dhcp pool** • **show ipv6 dhcp interface** **Example:** `Device# show ipv6 dhcp pool` or `Device# show ipv6 dhcp interface`	• Verifies DHCPv6 pool configuration. • Verifies that the DHCPv6 server function is enabled on an interface.
Step 14	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Enabling DHCPv6 Client Function

To enable the DHCPv6 client on an interface, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode, and specifies the interface to configure.

	Command or Action	Purpose
Step 4	**ipv6 address dhcp** [**rapid-commit**] **Example:** `Device(config-if)# ipv6 address dhcp rapid-commit`	Enables the interface to acquire an IPv6 address from the DHCPv6 server. **rapid-commit**—(Optional) Allow two-message exchange method for address assignment.
Step 5	**ipv6 dhcp client request** [**vendor-specific**] **Example:** `Device(config-if)# ipv6 dhcp client request vendor-specific`	(Optional) Enables the interface to request the vendor-specific option.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show ipv6 dhcp interface** **Example:** `Device# show ipv6 dhcp interface`	Verifies that the DHCPv6 client is enabled on an interface.

Enabling the Cisco IOS DHCP Server Database

For procedures to enable and configure the Cisco IOS DHCP server database, see the "DHCP Configuration Task List" section in the "Configuring DHCP" chapter of the Cisco IOS IP Configuration Guide.

Enabling the DHCP Snooping Binding Database Agent

Beginning in privileged EXEC mode, follow these steps to enable and configure the DHCP snooping binding database agent on the switch:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ip dhcp snooping database** {**flash**[*number*]**:**/*filename* | **ftp:**//*user***:***password***@***host*/*filename* | **http:**//[[*username***:***password*]**@**]{*hostname* | *host-ip*}[/*directory*] /*image-name***.tar** | **rcp:**//*user***@***host*/*filename*}| **tftp:**//*host*/*filename* **Example:** `Device(config)# ip dhcp snooping database tftp://10.90.90.90/snooping-rp2`	Specifies the URL for the database agent or the binding file by using one of these forms: • **flash**[*number*]**:**/*filename* • **ftp:**//*user***:***password***@***host*/*filename* • **http:**//[[*username***:***password*]**@**]{*hostname* | *host-ip*}[/*directory*] /*image-name***.tar** • **rcp:**//*user***@***host*/*filename* • **tftp:**//*host*/*filename*
Step 4	**ip dhcp snooping database timeout** *seconds* **Example:** `Device(config)# ip dhcp snooping database timeout 300`	Specifies (in seconds) how long to wait for the database transfer process to finish before stopping the process. The default is 300 seconds. The range is 0 to 86400. Use 0 to define an infinite duration, which means to continue trying the transfer indefinitely.
Step 5	**ip dhcp snooping database write-delay** *seconds* **Example:** `Device(config)# ip dhcp snooping database write-delay 15`	Specifies the duration for which the transfer should be delayed after the binding database changes. The range is from 15 to 86400 seconds. The default is 300 seconds (5 minutes).
Step 6	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 7	**ip dhcp snooping binding** *mac-address* **vlan** *vlan-id ip-address* **interface** *interface-id* **expiry** *seconds* **Example:** `Device# ip dhcp snooping binding 0001.1234.1234 vlan 1 172.20.50.5 interface gigabitethernet 1/1 expiry 1000`	(Optional) Adds binding entries to the DHCP snooping binding database. The *vlan-id* range is from 1 to 4904. The *seconds* range is from 1 to 4294967295. Enter this command for each entry that you add. Use this command when you are testing or debugging the switch.
Step 8	**show ip dhcp snooping database** [**detail**] **Example:**	Displays the status and statistics of the DHCP snooping binding database agent.

	Command or Action	Purpose
	`Device# show ip dhcp snooping database detail`	

Monitoring DHCP Snooping Information

Table 57: Commands for Displaying DHCP Information

Command	Description
show ip dhcp snooping	Displays the DHCP snooping configuration for a switch
show ip dhcp snooping binding	Displays only the dynamically configured bindings in the DHCP snooping binding database, also referred to as a binding table.
show ip dhcp snooping database	Displays the DHCP snooping binding database status and statistics.
show ip dhcp snooping statistics	Displays the DHCP snooping statistics in summary or detail form.
show ip source binding	Display the dynamically and statically configured bindings.

Note If DHCP snooping is enabled and an interface changes to the down state, the switch does not delete the statically configured bindings.

Enabling DHCP Server Port-Based Address Allocation

Follow these steps to globally enable port-based address allocation and to automatically generate a subscriber identifier on an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip dhcp use subscriber-id client-id** **Example:** `Device(config)# ip dhcp use subscriber-id client-id`	Configures the DHCP server to globally use the subscriber identifier as the client identifier on all incoming DHCP messages.
Step 4	**ip dhcp subscriber-id interface-name** **Example:** `Device(config)# ip dhcp subscriber-id interface-name`	Automatically generates a subscriber identifier based on the short name of the interface. A subscriber identifier configured on a specific interface takes precedence over this command.
Step 5	**interface** *interface-type interface-number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be configured, and enters interface configuration mode.
Step 6	**ip dhcp server use subscriber-id client-id** **Example:** `Device(config-if)# ip dhcp server use subscriber-id client-id`	Configures the DHCP server to use the subscriber identifier as the client identifier on all incoming DHCP messages on the interface.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

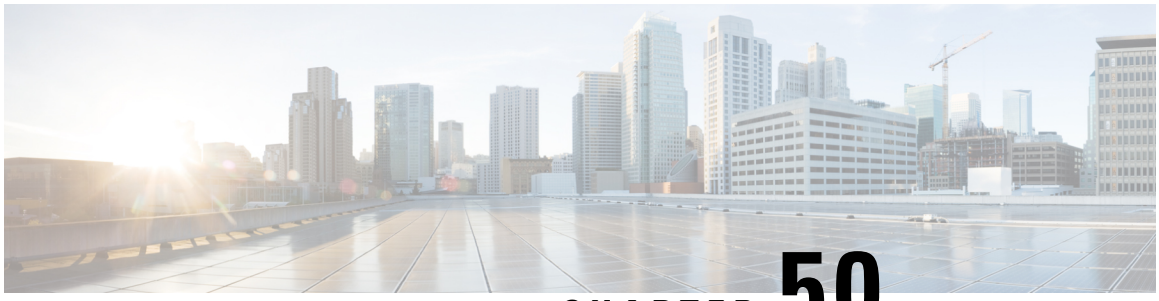
What to do next

After enabling DHCP port-based address allocation on the switch, use the **ip dhcp pool** global configuration command to preassign IP addresses and to associate them to clients.

Monitoring DHCP Server Port-Based Address Allocation

Table 58: Commands for Displaying DHCP Port-Based Address Allocation Information

Command	Purpose
show interface *interface id*	Displays the status and configuration of a specific interface.
show ip dhcp pool	Displays the DHCP address pools.
show ip dhcp binding	Displays address bindings on the Cisco IOS DHCP server.



CHAPTER 50

DHCP Gleaning

This section provides information about DHCP Gleaning.

Prerequisites for DHCP Gleaning

- Ensure that the interface to be configured is a Layer 2 interface.
- Ensure that global snooping is enabled.

Information About DHCP Gleaning

The following sections provide information about DHCP gleaning.

Overview of DHCP Gleaning

Gleaning helps extract location information from Dynamic Host Configuration Protocol (DHCP) messages when messages are forwarded by a DHCP relay agent; the process is a completely passive snooping functionality that neither blocks nor modifies DHCP packets. Additionally, gleaning helps to differentiate an untrusted device port that is connected to an end user from a trusted port connected to a DHCP server.

DHCP gleaning is a read–only DHCP snooping functionality that allows components to register and glean only DHCP version 4 packets. When you enable DHCP gleaning, it does a read-only snooping on all active interfaces on which DHCP snooping is disabled. You can add a secondary VLAN to a private VLAN. When add a secondary VLAN to a private VLAN, ensure that gleaning is enabled on the secondary VLAN, even though snooping is disabled on the primary VLAN. By default, the gleaning functionality is disabled. However, when you enable a device sensor, DHCP gleaning is automatically enabled.

DHCP Snooping

Dynamic Host Configuring Protocol (DHCP) snooping is a security feature that acts like a firewall between untrusted hosts and trusted DHCP servers. The DHCP snooping feature performs the following activities:

- Validates DHCP messages received from untrusted sources and filters out invalid messages.
- Rate-limits DHCP traffic from trusted and untrusted sources.
- Builds and maintains the DHCP snooping binding database, which contains information about untrusted hosts with leased IP addresses.
- Utilizes the DHCP snooping binding database to validate subsequent requests from untrusted hosts.

Other security features, such as dynamic Address Resolution Protocol (ARP) inspection (DAI), also uses information stored in the DHCP snooping binding database.

DHCP snooping is enabled on a per-VLAN basis. By default, the feature is inactive on all VLANs. You can enable the feature on a single VLAN or on a range of VLANs.

Configuring an Interface as a Trusted or an Untrusted Source for DHCP Gleaning

You can enable or disable DHCP gleaning on a device. You can configure an interface as a trusted or untrusted source of DHCP messages. Verify that no DHCP packets are dropped when DHCP gleaning is enabled on an untrusted interface or on a device port.

Note By default, DHCP gleaning is disabled.

You can configure DHCP trust on the following types of interfaces:

- Layer 2 Ethernet interfaces
- Layer 2 port-channel interfaces

Note By default, all interfaces are untrusted.

Procedure

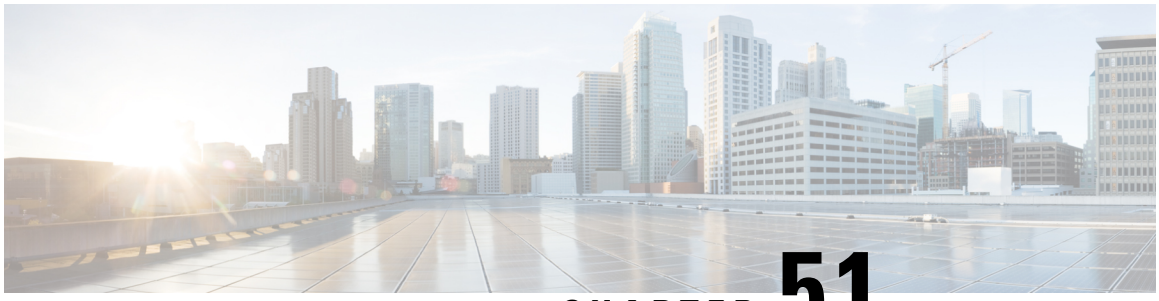
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip dhcp snooping glean** **Example:** `Device(config)# ip dhcp snooping glean`	Enables DHCP gleaning on an interface.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitEthernet 1/1`	Enters interface configuration mode, where *type number* is the Layer 2 Ethernet interface which you want to configure as trusted or untrusted for DHCP snooping.
Step 5	[**no**] **ip dhcp snooping trust** **Example:** `Device(config-if)# ip dhcp snooping trust`	Configures the interface as a trusted interface for DHCP snooping. The **no** option configures the port as an untrusted interface.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 7	**show ip dhcp snooping statistics** **Example:** `Device# show ip dhcp snooping statistics`	Displays packets that were dropped on the device port configured as an untrusted interface.
Step 8	**show ip dhcp snooping** **Example:** `Device# show ip dhcp snooping`	Displays DHCP snooping configuration information, including information about DHCP gleaning.

Example: Configuring an Interface as a Trusted or an Untrusted Source for DHCP Gleaning

This example shows how to enable Dynamic Host Configuration Protocol (DHCP) gleaning and configure an interface as a trusted interface:

```
Device> enable
Device# configure terminal
Device(config)# ip dhcp snooping glean
Device(config)# interface gigabitEthernet 1/1
Device(config-if)# ip dhcp snooping trust
Device(config-if)# end
```



CHAPTER 51

DHCP Options Support

Restrictions for DHCP Options Support

When DHCP snooping is configured on a primary VLAN, you cannot configure snooping with different settings on any of its secondary VLANs. You must configure DHCP snooping for all associated VLANs on the primary VLAN. If DHCP snooping is not configured on the primary VLAN and you try to configure it on the secondary VLAN, for example, VLAN 200, this message appears:

```
2w5d:%DHCP_SNOOPING-4-DHCP_SNOOPING_PVLAN_WARNING:DHCP Snooping configuration may not take
 effect
on secondary vlan 200. DHCP Snooping configuration on secondary vlan is derived from its
primary vlan.
```

You can use the **show ip dhcp snooping** command to display all VLANs, both primary and secondary, that have DHCP snooping enabled.

Information About DHCP Options Support

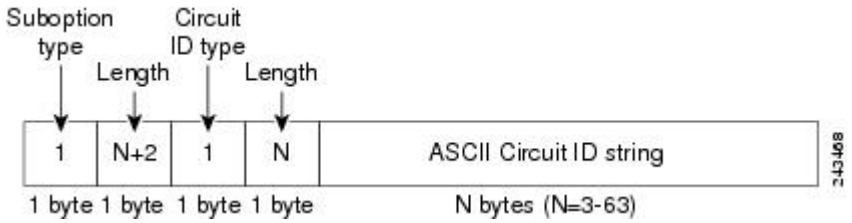
DHCP Option 82 Configurable Circuit ID and Remote ID Overview

The DHCP Option 82 Configurable Circuit ID and Remote ID feature enhances validation security by allowing you to determine what information is provided in the Option 82 Remote ID and Option 82 Circuit ID suboptions.

You can enable DHCP snooping on private VLANs. When DHCP snooping is enabled, the configuration is propagated to both a primary VLAN and its associated secondary VLANs. When DHCP snooping is enabled on a primary VLAN, it is also enabled on its secondary VLANs.

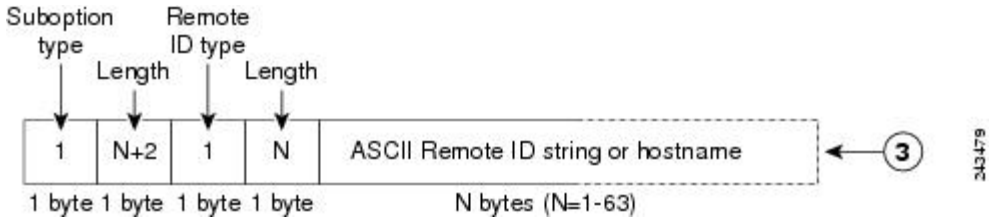
The figure below shows the packet format used when DHCP snooping is globally enabled and the **ip dhcp snooping information option** global configuration command is entered with the Circuit ID suboption.

Figure 57: Suboption Packet Formats, Circuit ID Specified



The figure below shows the packet format used when DHCP snooping is globally enabled and the **ip dhcp snooping information option** global configuration command is entered with the Remote ID suboption.

Figure 58: Suboption Packet Formats, Remote ID Specified



DHCP Client Option 12

The DHCP Client Option12 feature specifies the hostname of the client. While acquiring an IP address for an interface from the Dynamic Host Configuration Protocol (DHCP) server, if the client device receives the DHCP Hostname option inside the response, the hostname from that option is set. DHCP is used by DHCP clients to obtain configuration information for operation in an IP network.

Configuration parameters and other control information are carried in tagged data items that are stored in the options field of a DHCP message. The DHCP client provides flexibility by allowing Option 12 to be configured for a DHCP client.

Option 12 specifies the name of the client. The name might or might not be qualified with the local domain.

Configuring DHCP Snooping on Private VLANs

Perform these tasks to configure DHCP snooping on private primary and secondary VLANs:

- Configure a private, primary VLAN.
- Associate with it an isolated VLAN.
- Create an SVI interface for the primary VLAN, and associate it with the appropriate loopback IP and helper address.
- Enable DHCP snooping on the primary VLAN, which also enables it on the associated VLAN.

Note You must also configure a server to assign the IP address, a DHCP pool, and a relay route so that snooping can be effective.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan** *vlan-id* **Example:** `Device(config)# vlan 70`	Enters VLAN configuration mode for the named private VLAN.
Step 4	**private-vlan primary** **Example:** `Device(config-vlan)# private-vlan primary`	Designates the VLAN as the primary private VLAN.
Step 5	**private-vlan association** *secondary-vlan-list* **Example:** `Device(config-vlan)# private-vlan association 7`	Configures private VLANs (PVLANs) and the association between a PVLAN and a secondary VLAN.
Step 6	**exit** **Example:** `Device(ocnfig-vlan)# exit`	Exits VLAN configuration mode and returns to global configuration mode.
Step 7	**vlan** *vlan_ID* **Example:** `Device(config)# vlan 7`	Enters VLAN configuration mode for the named private VLAN. • In this example, the associated secondary VLAN is vlan 7.
Step 8	**private-vlan isolated** **Example:** `Device(config-vlan)# private-vlan isolated`	Designates the VLAN as an isolated private VLAN.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config-vlan)# exit`	Exits VLAN configuration mode and returns to global configuration mode.
Step 10	**interface vlan** *primary-vlan_id* **Example:** `Device(config)# interface vlan 70`	Creates a dynamic Switch Virtual Interface (SVI) on the primary VLAN, and enters interface configuration mode.
Step 11	**ip unnumbered loopback** **Example:** `Device(config-if)# ip unnumbered loopback1`	Specifies IP unnumbered loopback.
Step 12	**private-vlan mapping** [*secondary-vlan-list* | **add** *secondary-vlan-list*| **remove** *secondary-vlan-list*] **Example:** `Device(config-if)# private-vlan mapping 7`	Creates a mapping between the primary and the secondary VLANs so that they share the same primary VLAN SVI.
Step 13	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 14	**ip dhcp snooping vlan** *primary-vlan_id* **Example:** `Device(config)# ip dhcp snooping vlan 70`	Enables DHCP snooping on the primary and associated VLANs.
Step 15	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Example: Mapping Private-VLAN Associations

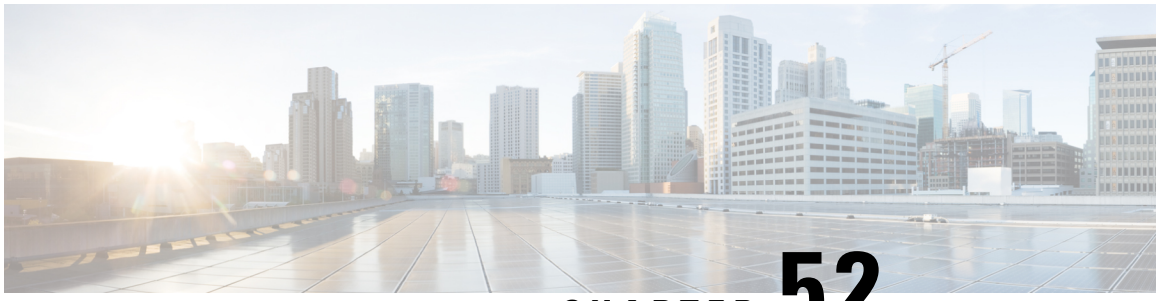
The following interface configuration example shows how to map the private-VLAN associations. The user-configurable circuit ID “aabb11” is inserted on the secondary VLAN, vlan 7.

```
Device> enable
Device# configure terminal
```

```
Device(config-if)# interface GigabitEthernet 1/1
Device(config-if)# switchport
Device(config-if)# switchport private-vlan host-association 70 7
Device(config-if)# switchport mode private-vlan host
Device(config-if)# spanning-tree portfast
Device(config-if)# exit
Device(config)# ip dhcp snooping vlan 7 information option format-type circuit-id string
aabb11
Device(config)# end
```

The following example shows how to define a DHCP class "C1" and specify the hex string of the corresponding class at the server by using the hex string that matches the circuit-ID value entered in the interface configuration example. That is, the hex string 00000000000000000000000000000006616162623131 mask ffffffffffffffffffffffffffff0000000000000 matches the circuit ID aabb11.

```
Device> enable
Device# configure terminal
Device(config)# ip dhcp class C1
Device(config-dhcp-class)# relay agent information
Device(config-dhcp-class-relayinfo)# relay-information hex
00000000000000000000000000000006616162623131
mask ffffffffffffffffffffffffffff0000000000000
Device(config-dhcp-class-relayinfo)# end
```



CHAPTER 52

DHCPv6 Options Support

Information About DHCPv6 Options Support

CAPWAP Access Controller DHCPv6 Option

The Control And Provisioning of Wireless Access Points (CAPWAP) protocol allows lightweight access points to use DHCPv6 to discover a wireless controller to which it can connect. CAPWAP is a standard, interoperable protocol that enables a controller to manage a collection of wireless access points.

Wireless access points use the DHCPv6 option 52 (RFC 5417) to supply the IPv6 management interface addresses of the primary, secondary, and tertiary wireless controllers.

Both stateless and stateful DHCPv6 addressing modes are supported. In stateless mode, access points obtain IPv6 address using the Stateless Address Auto Configuration (SLAAC), while additional network information (not obtained from router advertisements) is obtained from a DHCPv6 server. In stateful mode, access points obtain both IPv6 addressing and additional network information exclusively from the DHCPv6 server. In both modes, a DHCPv6 server is required to provide option 52 if Wireless Controller discovery using DHCPv6 is required.

When the MAX_PACKET_SIZE exceeds 15, and option 52 is configured, the DHCPv6 server does not send DHCP packets.

DNS Search List Option

DNS Search List (DNSSL) is a list of Domain Name System (DNS) suffix domain names used by IPv6 hosts when they perform DNS query searches for short, unqualified domain names. The DNSSL option contains one or more domain names. All domain names share the same lifetime value, which is the maximum time in seconds over which this DNSSL may be used. If different lifetime values are required, multiple DNSSL options can be used. There can be a maximum of 5 DNSSLs.

DHCP messages with long DNSSL names are discarded by the device.

Note If DNS information is available from multiple Router Advertisements (RAs) and/or from DHCP, the host must maintain an ordered list of this DNS information.

RFC 6106 specifies IPv6 Router Advertisement (RA) options to allow IPv6 routers to advertise a DNS Search List (DNSSL) to IPv6 hosts for an enhanced DNS configuration.

The DNS lifetime range should be between the maximum RA interval and twice the maximum RA interval, as displayed in the following example:

```
(max ra interval) <= dns lifetime <= (2*(max ra interval))
```

The maximum RA interval can have a value between 4 and 1800 seconds (the default is 240 seconds). The following example shows an out-of-range lifetime:

```
Device(config-if)# ipv6 nd ra dns-search-list sss.com 3600
! Lifetime configured out of range for the interface that has the  default maximum RA
interval.!
```

DHCPv6 Client Link-Layer Address Option

The DHCPv6 Client Link-Layer Address Option (RFC 6939) defines an optional mechanism and the related DHCPv6 option to allow first-hop DHCPv6 relay agents (relay agents that are connected to the same link as the client) to provide the client's link-layer address in DHCPv6 messages that are sent towards the server.

The Client Link-Layer Address option is only exchanged between relay agents and servers. DHCPv6 clients are not aware of the use of the Client Link-Layer Address option. The DHCPv6 client must not send the Client Link-Layer Address option, and must ignore the Client Link-Layer Address option if received.

Each DHCPv6 client and server is identified by a DHCP unique identifier (DUID). The DUID is carried in the client identifier and server identifier options. The DUID is unique across all DHCP clients and servers, and it is stable for any specific client or server. DHCPv6 uses DUIDs based on link-layer addresses for both the client and server identifier. The device uses the MAC address from the lowest-numbered interface to form the DUID. The network interface is assumed to be permanently attached to the device.

DHCP Relay Agent

A DHCP relay agent is a Layer 3 device that forwards DHCP packets between clients and servers. Relay agents forward requests and replies between clients and servers when they are not on the same physical subnet. Relay agent forwarding is different from the normal Layer 2 forwarding, in which IP datagrams are switched transparently between networks. Relay agents receive DHCP messages and generate new DHCP messages to send on output interfaces.

DHCPv6 Relay Agent

A DHCPv6 relay agent, which may reside on the client’s link, is used to relay messages between the client and the server. The DHCPv6 relay agent operation is transparent to the client. A DHCPv6 client locates a DHCPv6 server using a reserved, link-scoped multicast address. For direct communication between the DHCPv6 client and the DHCPv6 server, both of them must be attached to the same link. However, in some

situations where ease of management, economy, or scalability is a concern, it is desirable to allow a DHCPv6 client to send a message to a DHCPv6 server that is not connected to the same link.

DHCPv6 Relay Interface-Id Option

A DHCPv6 relay agent adds an Interface-Id option in the upstream DHCPv6 message. The Interface-Id option serves to identify the interface on which the client is connected. This information is used by the DHCPv6 relay agent while forwarding the downstream DHCPv6 message to the DHCPv6 client.

In a scenario where a Switch Virtual Interface (SVI) is configured to act as a relay agent, the Interface-Id option does not carry the physical interface details of the client interface. The Interface-Id option contains only the VLAN number of the client interface. The DHCPv6 server cannot identify which client sent the packet. The server cannot assign IPv6 addressess and policies to the packet.

When an SVI acts as a relay agent the Interface-Id option will contain the physical interface details of the client interface. The physical interface details are included along with the VLAN number which is included by default. The new data is added as a sub-option. This makes it backward compatible as well as easily extensible.

The following is an example of the Interface-Id format before the physical interface details of the client interface are included.

```
Interface-Id String: 0x0105566C313030
Sub-op code: 01
Length :05
data: 566C313030 (Vlan100)
```

The following is an example of the Interface-Id format after the physical interface details of the client interface are included.

```
Interface-Id String: 0x0105566C31303002074769302F312F30
New sub option to include physical interface name
Sub-op code: 02
length:07
data:4769302F312F30 (Gi1/1)
```

How to Configure DHCPv6 Options Support

This section provides information about how to configure DHCPv6 options support:

Configuring CAPWAP Access Points

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Device(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 server configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**capwap-ac address** *ipv6-address* **Example:** `Device(config-dhcpv6)# capwap-ac address 2001:DB8::1`	Configures CAPWAP access controller address.
Step 5	**end** **Example:** `Device(config-dhcpv6)# end`	Exits DHCPv6 pool configuration mode and returns to privileged EXEC mode.

Configuring DNS Search List Using IPv6 Router Advertisement Options

Perform this task to configure the DNS search list using IPv6 router advertisement options:

Note The domain name configuration should follow RFC 1035. If not, the configuration will be rejected. For example, the following domain name configuration will result in an error:

```
Device(config-if)# ipv6 nd ra dns-search-list domain example.example.com infinite-lifetime
```

Note The **ipv6 nd ra dns-search-list domain** command can only be configured on physical interfaces that are configured as routed ports in layer 3 mode. This is done by running the **no switchport** command in interface configuration mode.

Use the **no ipv6 nd ra dns-search-list domain** *domain-name* command in interface configuration mode to delete a single DNS search list under an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-type interface-number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Configures an interface and enters interface configuration mode.
Step 4	**no switchport** **Example:** `Device(config-if)# no switchport`	For physical ports only, enters Layer 3 mode.
Step 5	**ipv6 nd prefix** *ipv6-prefix/prefix-length* **Example:** `Device(config-if)# ipv6 nd prefix 2001:DB8::1/64 1111 222`	Configures IPv6 prefixes that are included in IPv6 Neighbor Discovery (ND) router advertisements.
Step 6	**ipv6 nd ra lifetime** *seconds* **Example:** `Device(config-if)# ipv6 nd ra lifetime 9000`	Configures the device lifetime value in IPv6 router advertisements on an interface.
Step 7	**ipv6 nd ra dns-search-list domain** *domain-name* [**lifetime** [*lifetime-value* | **infinite**]] **Example:** `Device(config-if)# ipv6 nd ra dns-search-list domain example.example.com lifetime infinite`	Configures the DNS search list. You can specify the life time of the search list.
Step 8	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example: Configuring CAPWAP Access Points

The following example shows how to configure a CAPWAP access point:

```
Device> enable
Device# configure terminal
Device(config)# ipv6 dhcp pool pool1
Device(config-dhcpv6)# capwap-ac address 2001:DB8::1
Device(config-dhcpv6)# end
Device#
```

Verifying DHCPv6 Options Support

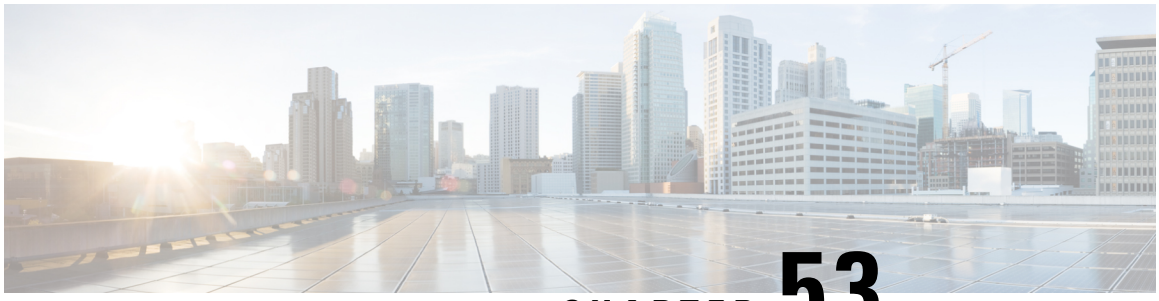
Verifying Option 52 Support

The following sample output from the **show ipv6 dhcp pool** command displays the DHCPv6 configuration pool information:

```
Device# show ipv6 dhcp pool

DHCPv6 pool: svr-p1
  Static bindings:
    Binding for client 000300010002FCA5C01C
      IA PD: IA ID 00040002,
        Prefix: 2001:db8::3/72
                preferred lifetime 604800, valid lifetime 2592000
      IA PD: IA ID not specified; being used by 00040001
        Prefix: 2001:db8::1/72
                preferred lifetime 240, valid lifetime 54321
        Prefix: 2001:db8::2/72
                preferred lifetime 300, valid lifetime 54333
        Prefix: 2001:db8::3/72
                preferred lifetime 280, valid lifetime 51111
  Prefix from pool: local-p1, Valid lifetime 12345, Preferred lifetime 180
  DNS server: 1001::1
  DNS server: 1001::2
  CAPWAP-AC Controller address: 2001:DB8::1
  Domain name: example1.com
  Domain name: example2.com
  Domain name: example3.com
 Active clients: 2
```

The following example shows how to enable debugging for DHCPv6:

```
Device# debug ipv6 dhcp detail

IPv6 DHCP debugging is on (detailed)
```



CHAPTER 53

DHCPv6 Relay Source Configuration

Restrictions for Configuring a DHCPv6 Relay Source

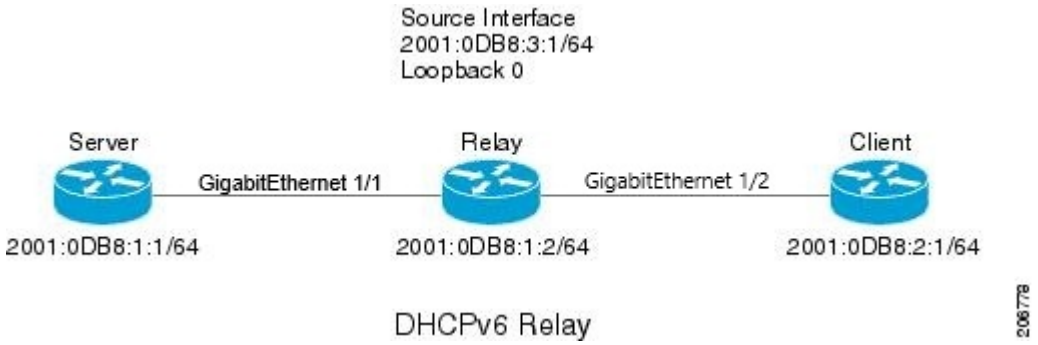
- If the configured interface is shut down, or if all of its IPv6 addresses are removed, the relay will revert to its standard behavior.
- The command line interface (CLI) will report an error if the user attempts to specify an interface that has no IPv6 addresses configured.
- The interface configuration takes precedence over the global configuration if both have been configured.

Information About DHCPv6 Relay Source Configuration

The DHCPv6 server sends its replies to the source address of relayed messages. Normally, a DHCPv6 relay uses the address of the server-facing interface used to send messages as the source. However, in some networks, it may be desirable to configure a more stable address (such as a loopback interface) and have the relay use that interface as the source address of relayed messages. The DHCPv6 Relay Source Configuration feature provides this capability.

The figure below shows a simple network with a single client, relay, and server. The relay and server communicate over 2001:DB8:1::/64, and the relay has a client-facing interface on 2001:DB8:2::/64. The relay also has a loopback interface configured with address 2001:DB8:3:1/64.

Figure 59: DHCPv6 Relay Source Configuration—Simple Network



When the relay receives a request from the client, the relay includes an address from the client-facing interface (GigabitEthernet 1/2) in the link-address field of a relay-forward message. This address is used by the server to select an address pool. The relay then sends the relay-forward message toward the server. By default, the address of the server-facing (GigabitEthernet 1/1) interface is used as the IPv6 source, and the server will send any reply to that address.

If the relay source interface is explicitly configured, the relay will use that interface's primary IPv6 address as the IPv6 source for messages it forwards. For example, configuring Loopback 0 as the source would cause the relay to use 2001:DB8:3:1/64 as the IPv6 source address for messages relayed toward the server.

Configuring a DHCPv6 Relay Source

Perform the following tasks to configure a DHCPv6 relay source:

Configuring a DHCPv6 Relay Source on an Interface

Perform this task to configure an interface to use as the source when relaying messages.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface loopback 0`	Specifies an interface type and number, and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ipv6 dhcp relay source-interface** *interface-type interface-number* **Example:** `Device(config-if)# ipv6 dhcp relay source-interface loopback 0`	Configures an interface to use as the source when relaying messages received on this interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring a DHCPv6 Relay Source Globally

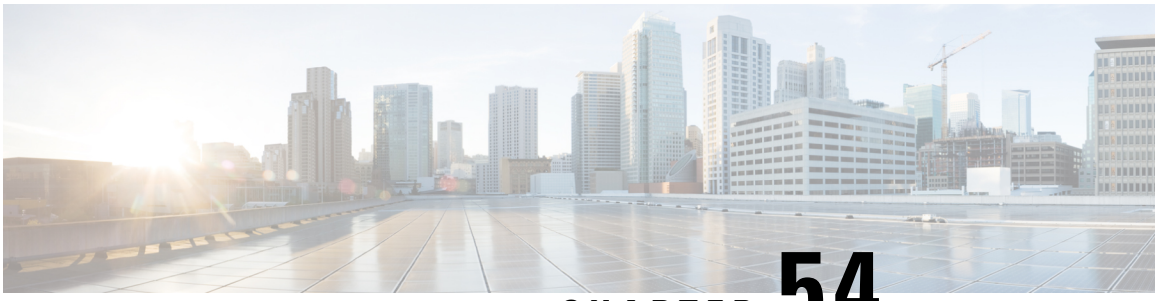
Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp relay source-interface** *interface-type interface-number* **Example:** `Device(config)# ipv6 dhcp relay source-interface loopback 0`	Configures an interface to use as the source when relaying messages.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Example: Configuring a DHCPv6 Relay Source on an Interface

The following example show how to configure the Loopback 0 interface to be used as the relay source:

```
Device> enable
Device# configure terminal
Device(config)# interface loopback 0
Device(config-if)# ipv6 dhcp relay source-interface loopback 0
Device(config-if)# end
```



CHAPTER 54

Configuring IPv6 over IPv4 GRE Tunnels

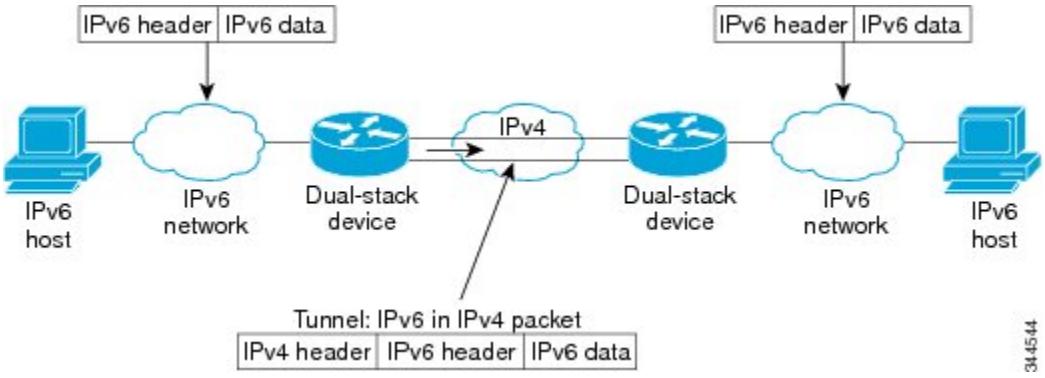
Information About Configuring IPv6 over IPv4 GRE Tunnels

The following sections provide information about configuring IPv6 over IPv4 GRE tunnels:

Overlay Tunnels for IPv6

Overlay tunneling encapsulates IPv6 packets in IPv4 packets for delivery across an IPv4 infrastructure (a core network or the figure below). By using overlay tunnels, you can communicate with isolated IPv6 networks without upgrading the IPv4 infrastructure between them. Overlay tunnels can be configured between border devices or between a border device and a host; however, both tunnel endpoints must support both the IPv4 and IPv6 protocol stacks.

Figure 60: Overlay Tunnels



Note Overlay tunnels reduce the maximum transmission unit (MTU) of an interface by 20 octets (assuming that the basic IPv4 packet header does not contain optional fields). A network that uses overlay tunnels is difficult to troubleshoot. Therefore, overlay tunnels that connect isolated IPv6 networks should not be considered a final IPv6 network architecture. The use of overlay tunnels should be considered as a transition technique toward a network that supports both the IPv4 and IPv6 protocol stacks or just the IPv6 protocol stack.

IPv6 supports GRE type of overlay tunneling. IPv6 over IPv4 GRE Tunnels can carry IPv6, Connectionless Network Service (CLNS), and many other types of packets.

GRE IPv4 Tunnel Support for IPv6 Traffic

IPv6 traffic can be carried over IPv4 GRE tunnels using the standard GRE tunneling technique that is designed to provide the services to implement any standard point-to-point encapsulation scheme. As in IPv6 manually configured tunnels, GRE tunnels are links between two points, with a separate tunnel for each link. The tunnels are not tied to a specific passenger or transport protocol but, in this case, carry IPv6 as the passenger protocol with the GRE as the carrier protocol and IPv4 or IPv6 as the transport protocol.

The primary use of GRE tunnels is for stable connections that require regular secure communication between two edge devices or between an edge device and an end system. The edge devices and the end systems must be dual-stack implementations.

Configuring GRE IPv6 Tunnels

Perform this task to configure a GRE tunnel on an IPv6 network. GRE tunnels can be configured to run over an IPv6 network layer and to transport IPv6 packets in IPv6 tunnels and IPv4 packets in IPv6 tunnels.

To configure GRE IPv6 tunnels, perform this procedure:

Before you begin

When GRE IPv6 tunnels are configured, IPv6 addresses are assigned to the tunnel source and the tunnel destination. The tunnel interface can have either IPv4 or IPv6 addresses assigned (this is not shown in the task). The host or router at each end of a configured tunnel must support both the IPv4 and IPv6 protocol stacks.

Procedure

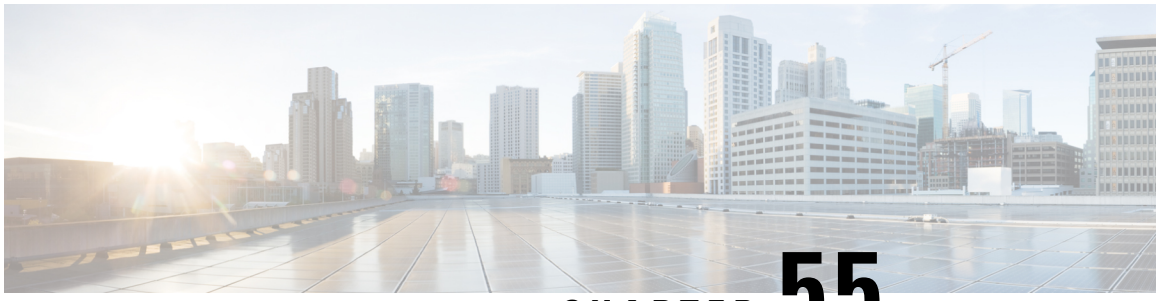
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**interface tunnel** *tunnel-number* **Example:** `Device(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-prefix* / *prefix-length* [**eui-64**] **Example:** `Device(config-if)# ipv6 address 3ffe:b00:c18:1::3/127`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface.
Step 5	**tunnel source** {*ip-address* | *ipv6-address* | *interface-type interface-number*} **Example:** `Device(config-if)# tunnel source ethernet 0`	Specifies the source IPv4 address or the source interface type and number for the tunnel interface. • If an interface is specified, the interface must be configured with an IPv4 address.
Step 6	**tunnel destination** {*host-name* | *ip-address* | *ipv6-address*} **Example:** `Device(config-if)# tunnel destination 2001:DB8:1111:2222::1/64`	Specifies the destination IPv6 address or hostname for the tunnel interface.
Step 7	**tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre**| **gre multipoint** | **gre ipv6** | **ipip** [**decapsulate-any**] | **iptalk** | **ipv6** | **mpls** | **nos** **Example:** `Device(config-if)# tunnel mode gre ipv6`	Specifies a GRE IPv6 tunnel. **Note** The **tunnel mode gre ipv6** command specifies GRE as the encapsulation protocol for the tunnel.

Configuration Example: Tunnel Destination Address for IPv6 Tunnel

```
Device> enable
Device# configure terminal
Device(config)# interface Tunnel 0
Device(config-if)# ipv6 address 2001:1:1::1/48
Device(config-if)# tunnel source GigabitEthernet 1/1
Device(config-if)# tunnel destination 10.0.0.2
Device(config-if)# tunnel mode gre ipv6
Device(config-if)# exit
```

```
!
Device(config)# interface GigabitEthernet1/1
Device(config-if)# ip address 10.0.0.1 255.255.255.0
Device(config-if)# exit
!
Device(config)# ipv6 unicast-routing
Device(config)# router isis
Device(config-router)# net 49.0000.0000.000a.00
```



CHAPTER 55

Configuring HSRP

Information About Hot Standby Router Protocol

The following sections provide information about Hot Standby Router Protocol (HSRP)

HSRP Overview

HSRP is Cisco's standard method of providing high network availability by providing first-hop redundancy for IP hosts on an IEEE 802 LAN configured with a default gateway IP address. HSRP routes IP traffic without relying on the availability of any single router. It enables a set of router interfaces to work together to present the appearance of a single virtual router or default gateway to the hosts on a LAN. When HSRP is configured on a network or segment, it provides a virtual Media Access Control (MAC) address and an IP address that is shared among a group of configured routers. HSRP allows two or more HSRP-configured routers to use the MAC address and IP network address of a virtual router. The virtual router does not exist; it represents the common target for routers that are configured to provide backup to each other. One of the routers is selected to be the active router and another to be the standby router, which assumes control of the group MAC address and IP address should the designated active router fail.

Note Routers in an HSRP group can be any router interface that supports HSRP, including routed ports and switch virtual interfaces (SVIs).

HSRP provides high network availability by providing redundancy for IP traffic from hosts on networks. In a group of router interfaces, the active router is the router of choice for routing packets; the standby router is the router that takes over the routing duties when an active router fails or when preset conditions are met.

HSRP is useful for hosts that do not support a router discovery protocol and cannot switch to a new router when their selected router reloads or loses power. When HSRP is configured on a network segment, it provides a virtual MAC address and an IP address that is shared among router interfaces in a group of router interfaces running HSRP. The router selected by the protocol to be the active router receives and routes packets destined for the group's MAC address. For n routers running HSRP, there are n +1 IP and MAC addresses assigned.

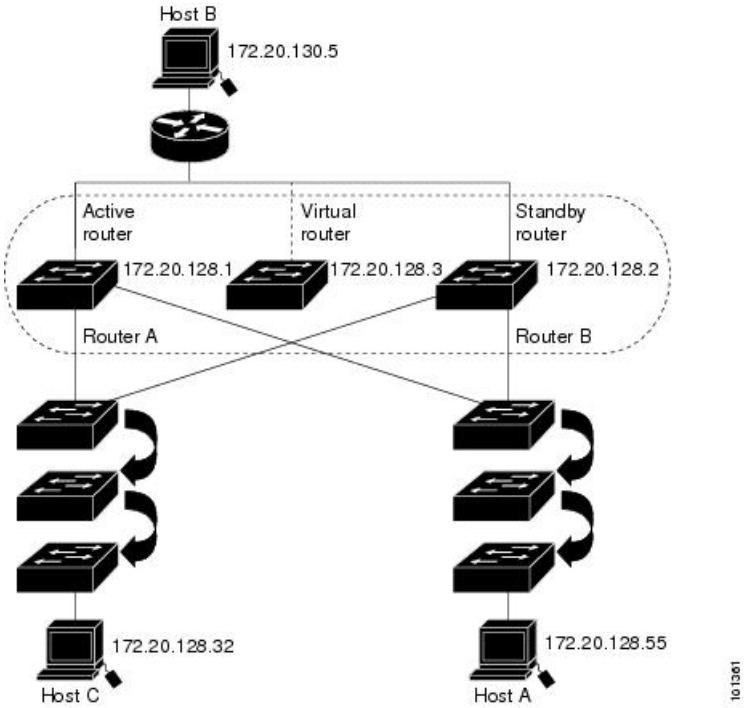
HSRP detects when the designated active router fails, and a selected standby router assumes control of the Hot Standby group's MAC and IP addresses. A new standby router is also selected at that time. Devices running HSRP send and receive multicast UDP-based hello packets to detect router failure and to designate active and standby routers. When HSRP is configured on an interface, Internet Control Message Protocol (ICMP) redirect messages are automatically enabled for the interface.

You can configure multiple Hot Standby groups among switches that are operating in Layer 3 to make more use of the redundant routers.

To do so, specify a group number for each Hot Standby command group you configure for an interface. For example, you might configure an interface on switch 1 as an active router and one on switch 2 as a standby router and also configure another interface on switch 2 as an active router with another interface on switch 1 as its standby router.

The following figure shows a segment of a network configured for HSRP. Each router is configured with the MAC address and IP network address of the virtual router. Instead of configuring hosts on the network with the IP address of Router A, you configure them with the IP address of the virtual router as their default router. When Host C sends packets to Host B, it sends them to the MAC address of the virtual router. If for any reason, Router A stops transferring packets, Router B responds to the virtual IP address and virtual MAC address and becomes the active router, assuming the active router duties. Host C continues to use the IP address of the virtual router to address packets destined for Host B, which Router B now receives and sends to Host B. Until Router A resumes operation, HSRP allows Router B to provide uninterrupted service to users on Host C's segment that need to communicate with users on Host B's segment and also continues to perform its normal function of handling packets between the Host A segment and Host B.

Figure 61: Typical HSRP Configuration



HSRP Versions

The switch supports these HSRP versions:

- HSRPv1- Version 1 of the HSRP, the default version of HSRP. It has these features:
  - The HSRP group number can be from 0 to 255.
  - HSRPv1 uses the multicast address 224.0.0.2 to send hello packets, which can conflict with Cisco Group Management Protocol (CGMP) leave processing. You cannot enable HSRPv1 and CGMP at the same time; they are mutually exclusive.
- HSRPv2- Version 2 of the HSRP has these features:
  - HSRPv2 uses the multicast address 224.0.0.102 to send hello packets. HSRPv2 and CGMP leave processing are no longer mutually exclusive, and both can be enabled at the same time.
  - HSRPv2 has a different packet format than HRSPv1.

A switch running HSRPv1 cannot identify the physical router that sent a hello packet because the source MAC address of the router is the virtual MAC address.

HSRPv2 has a different packet format than HSRPv1. A HSRPv2 packet uses the type-length-value (TLV) format and has a 6-byte identifier field with the MAC address of the physical router that sent the packet.

If an interface running HSRPv1 gets an HSRPv2 packet, the type field is ignored.

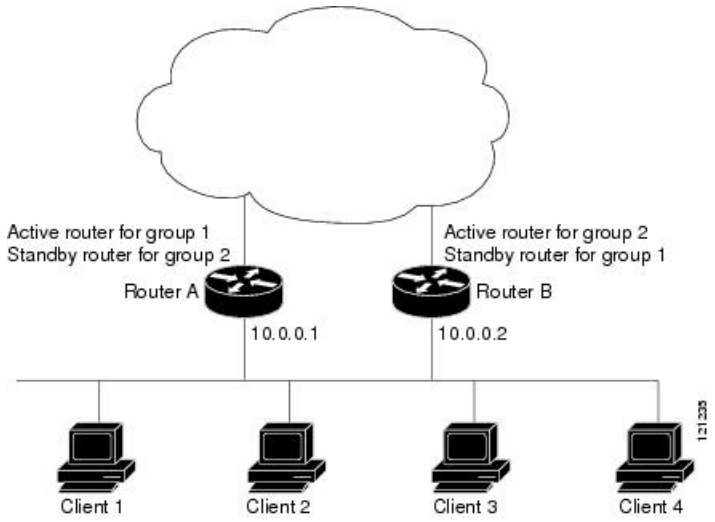
Multiple HSRP

The switch supports Multiple HSRP (MHSRP), an extension of HSRP that allows load sharing between two or more HSRP groups. You can configure MHSRP to achieve load-balancing and to use two or more standby groups (and paths) from a host network to a server network.

In the figure below, half the clients are configured for Router A, and half the clients are configured for Router B. Together, the configuration for Routers A and B establishes two HSRP groups. For group 1, Router A is the default active router because it has the assigned highest priority, and Router B is the standby router. For group 2, Router B is the default active router because it has the assigned highest priority, and Router A is the standby router. During normal operation, the two routers share the IP traffic load. When either router becomes unavailable, the other router becomes active and assumes the packet-transfer functions of the router that is unavailable.

Note For MHSRP, you need to enter the **standby preempt** interface configuration command on the HSRP interfaces so that if a router fails and then comes back up, preemption restores load sharing.

Figure 62: MHSRP Load Sharing



Configuring HSRP for IPv6

Switches running the support the Hot Standby Router Protocol (HSRP) for IPv6. HSRP provides routing redundancy for routing IPv6 traffic not dependent on the availability of any single router. IPv6 hosts learn of available routers through IPv6 neighbor discovery router advertisement messages. These messages are multicast periodically or are solicited by hosts.

An HSRP IPv6 group has a virtual MAC address that is derived from the HSRP group number and a virtual IPv6 link-local address that is, by default, derived from the HSRP virtual MAC address.

Periodic messages are sent for the HSRP virtual IPv6 link-local address when the HSRP group is active. These messages stop after a final one is sent when the group leaves the active state.

Note When configuring HSRP for IPv6, you must enable HSRP version 2 (HSRPv2) on the interface.

HSRP IPv6 Virtual MAC Address Range

HSRP IPv6 uses a different virtual MAC address block than does HSRP for IP:

0005.73A0.0000 through 0005.73A0.0FFF (4096 addresses)

HSRP IPv6 UDP Port Number

Port number 2029 has been assigned to HSRP IPv6.

How to Configure Hot Standby Router Protocol

The following sections provide configuration information about HSRP.

Default HSRP Configuration

Table 59: Default HSRP Configuration

Feature	Default Setting
HSRP version	Version 1
HSRP groups	None configured
Standby group number	0
Standby MAC address	System assigned as: 0000.0c07.acXX, where XX is the HSRP group number
Standby priority	100
Standby delay	0 (no delay)
Standby track interface priority	10
Standby hello time	3 seconds
Standby holdtime	10 seconds

HSRP Configuration Guidelines

- HSRPv2 and HSRPv1 are mutually exclusive. HSRPv2 is not interoperable with HSRPv1 on an interface and the reverse.
- In the procedures, the specified interface must be one of these Layer 3 interfaces:
  - Routed port: A physical port configured as a Layer 3 port by entering the **no switchport** command in interface configuration mode.
  - SVI: A VLAN interface created by using the **interface vlan** *vlan_id* in global configuration mode, and by default a Layer 3 interface.
  - Etherchannel port channel in Layer 3 mode: A port-channel logical interface created by using the **interface port-channel** *port-channel-number* in global configuration mode, and binding the Ethernet interface into the channel group.
- All Layer 3 interfaces must have IP addresses assigned to them.
- HSRP millisecond timers are not supported.

Enabling HSRP

The **standby ip** interface configuration command activates HSRP on the configured interface. If an IP address is specified, that address is used as the designated address for the Hot Standby group. If no IP address is specified, the address is learned through the standby function. You must configure at least one Layer 3 port on the LAN with the designated address. Configuring an IP address always overrides another designated address currently in use.

When the **standby ip** command is enabled on an interface and proxy ARP is enabled, if the interface's Hot Standby state is active, proxy ARP requests are answered using the Hot Standby group MAC address. If the interface is in a different state, proxy ARP responses are suppressed.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Enters interface configuration mode, and enter the Layer 3 interface on which you want to enable HSRP.
Step 3	**standby version** {**1** | **2**} **Example:** `Device(config-if)# standby version 1`	(Optional) Configures the HSRP version on the interface. • 1- Selects HSRPv1. • 2- Selects HSRPv2. If you do not enter this command or do not specify a keyword, the interface runs the default HSRP version, HSRP v1.
Step 4	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 1 ip`	Creates (or enable) the HSRP group using its number and virtual IP address. • (Optional) group-number- The group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) ip-address- The virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**- The IP address is a secondary hot standby router interface. If neither router is designated as a secondary or standby router and no priorities are set, the primary IP addresses are compared and the higher IP address is

	Command or Action	Purpose
		the active router, with the next highest as the standby router.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode
Step 6	**show standby** [*interface-id* [*group*]] **Example:** `Device# show standby`	Verifies the configuration of the standby groups.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Enabling and Verifying an HSRP Group for IPv6 Operation

In this task, when you enter the **standby ipv6** command, a link-local address is generated from the link-local prefix, and a modified EUI-64 format interface identifier is generated in which the EUI-64 interface identifier is created from the relevant HSRP virtual MAC address.

A link-local address is an IPv6 unicast address that can be automatically configured on any interface using the link-local prefix FE80::/10 (1111 1110 10) and the interface identifier in the modified EUI-64 format. Link-local addresses are used in the stateless autoconfiguration process. Nodes on a local link can use link-local addresses to communicate; the nodes do not need site-local or globally unique addresses to communicate.

In IPv6, a device on the link advertises in RA messages any site-local and global prefixes, and its willingness to function as a default device for the link. RA messages are sent periodically and in response to router solicitation messages, which are sent by hosts at system startup.

A node on the link can automatically configure site-local and global IPv6 addresses by appending its interface identifier (64 bits) to the prefixes (64 bits) included in the RA messages. The resulting 128-bit IPv6 addresses configured by the node are then subjected to duplicate address detection to ensure their uniqueness on the link. If the prefixes advertised in the RA messages are globally unique, then the IPv6 addresses configured by the node are also guaranteed to be globally unique. Router solicitation messages, which have a value of 133 in the Type field of the ICMP packet header, are sent by hosts at system startup so that the host can immediately autoconfigure without needing to wait for the next scheduled RA message.

To enabling and verifying an HSRP group for IPv6, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams. • The **ipv6 unicast-routing** command must be enabled for HSRP for IPv6 to work.
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 5	**standby** [*group-number*] **ipv6** {*link-local-address* | **autoconfig**} **Example:** `Device(config-if)# standby 1 ipv6 autoconfig`	Activates the HSRP in IPv6.
Step 6	**standby** [*group-number*] **preempt** [**delay minimum** *seconds* | **reload** *seconds* | **sync** *seconds*] **Example:** `Device(config-if)# standby 1 preempt`	Configures HSRP preemption and preemption delay.
Step 7	**standby** [*group-number*] **priority** *priority* **Example:** `Device(config-if)# standby 1 priority 110`	Configures HSRP priority.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Returns the device to privileged EXEC mode.
Step 9	**show standby** [*type number* [*group*]] [**all** | **brief**] **Example:**	Displays HSRP information.

	Command or Action	Purpose
	`Device# show standby`	
Step 10	**show ipv6 interface** [**brief**] [*interface-type interface-number*] [**prefix**] **Example:** `Device# show ipv6 interface GigabitEthernet 1/1`	Displays the usability status of interfaces configured for IPv6.

Configuring HSRP Priority

The **standby priority**, **standby preempt**, and **standby track** interface configuration commands are all used to set characteristics for finding active and standby routers and behavior regarding when a new active router takes over.

When configuring HSRP priority, follow these guidelines:

- Assigning a priority allows you to select the active and standby routers. If preemption is enabled, the router with the highest priority becomes the active router. If priorities are equal, the current active router does not change.
- The highest number (1 to 255) represents the highest priority (most likely to become the active router).
- When setting the priority, preempt, or both, you must specify at least one keyword (**priority**, **preempt**, or both)
- The priority of the device can change dynamically if an interface is configured with the **standby track** command and another interface on the router goes down.
- The **standby track** interface configuration command ties the router hot standby priority to the availability of its interfaces and is useful for tracking interfaces that are not configured for HSRP. When a tracked interface fails, the hot standby priority on the device on which tracking has been configured decreases by 10. If an interface is not tracked, its state changes do not affect the hot standby priority of the configured device. For each interface configured for hot standby, you can configure a separate list of interfaces to be tracked.
- The **standby track** *interface-priority* interface configuration command specifies how much to decrement the hot standby priority when a tracked interface goes down. When the interface comes back up, the priority is incremented by the same amount.
- When multiple tracked interfaces are down and *interface-priority* values have been configured, the configured priority decrements are cumulative. If tracked interfaces that were not configured with priority values fail, the default decrement is 10, and it is noncumulative.
- When routing is first enabled for the interface, it does not have a complete routing table. If it is configured to preempt, it becomes the active router, even though it is unable to provide adequate routing services. To solve this problem, configure a delay time to allow the router to update its routing table.

Beginning in privileged EXEC mode, use one or more of these steps to configure HSRP priority characteristics on an interface:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Enters interface configuration mode, and enter the HSRP interface on which you want to set priority.
Step 3	**standby** [*group-number*] **priority** *priority* **Example:** `Device(config-if)# standby 120 priority 50`	Sets a **priority** value used in choosing the active router. The range is 1 to 255; the default priority is 100. The highest number represents the highest priority. (Optional) group-number—The group number to which the command applies. Use the **no** form of the command to restore the default values.
Step 4	**standby** [*group-number*] **preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:** `Device(config-if)# standby 1 preempt delay 300`	Configures the router to **preempt**, which means that when the local router has a higher priority than the active router, it becomes the active router. • (Optional) group-number-The group number to which the command applies. • (Optional) **delay minimum**—Set to cause the local router to postpone taking over the active role for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). • (Optional) **delay reload**—Set to cause the local router to postpone taking over the active role after a reload for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over after a reload). • (Optional) **delay sync**—Set to cause the local router to postpone taking over the active role so that IP redundancy clients can reply (either with an ok or wait reply) for the number of seconds shown. The

	Command or Action	Purpose
		range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). Use the **no** form of the command to restore the default values.
Step 5	**standby** [*group-number*] **track** *type number* [*interface-priority*] **Example:** `Device(config-if)# standby track interface gigabitethernet1/1`	Configures an interface to track other interfaces so that if one of the other interfaces goes down, the device's Hot Standby priority is lowered. • (Optional) group-number- The group number to which the command applies. • type- Enter the interface type (combined with interface number) that is tracked. • number- Enter the interface number (combined with interface type) that is tracked. • (Optional) interface-priority- Enter the amount by which the hot standby priority for the router is decremented or incremented when the interface goes down or comes back up. The default value is 10.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config**	Verifies the configuration of the standby groups.
Step 8	**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Configuring MHSRP

To enable MHSRP and load-balancing, you configure two routers as active routers for their groups, with virtual routers as standby routers as shown in the *MHSRP Load Sharing* figure in the Multiple HSRP section. You need to enter the **standby preempt** interface configuration command on each HSRP interface so that if a router fails and comes back up, the preemption occurs and restores load-balancing.

Router A is configured as the active router for group 1, and Router B is configured as the active router for group 2. The HSRP interface for Router A has an IP address of 10.0.0.1 with a group 1 standby priority of 110 (the default is 100). The HSRP interface for Router B has an IP address of 10.0.0.2 with a group 2 standby priority of 110.

Group 1 uses a virtual IP address of 10.0.0.3 and group 2 uses a virtual IP address of 10.0.0.4.

Configuring Router A

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet1/1`	Configures an interface type and enters interface configuration mode.
Step 3	**no switchport** **Example:** `Device(config)# no switchport`	Switches an interface that is in Layer 2 mode into Layer 3 mode for Layer 3 configuration.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.0.0.1 255.255.255.0`	Specifies an IP address for an interface.
Step 5	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 1 ip 10.0.0.3`	Creates the HSRP group using its number and virtual IP address. • (Optional) *group-number*- The group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) *ip-address*- The virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**- The IP address is a secondary hot standby router interface. If neither router is designated as a secondary or standby router and no priorities are set, the primary IP addresses are compared and the higher IP address

	Command or Action	Purpose
		is the active router, with the next highest as the standby router.
Step 6	**standby** [*group-number*] **priority** *priority* **Example:** `Device(config-if)# standby 1 priority 110`	Sets a **priority** value used in choosing the active router. The range is 1 to 255; the default priority is 100. The highest number represents the highest priority. (Optional) *group-number*—The group number to which the command applies. Use the **no** form of the command to restore the default values.
Step 7	**standby** [*group-number*] **preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:** `Device(config-if)# standby 1 preempt delay 300`	Configures the router to **preempt**, which means that when the local router has a higher priority than the active router, it becomes the active router. • (Optional) group-number-The group number to which the command applies. • (Optional) **delay minimum**—Set to cause the local router to postpone taking over the active role for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). • (Optional) **delay reload**—Set to cause the local router to postpone taking over the active role after a reload for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over after a reload) . • (Optional) **delay sync**—Set to cause the local router to postpone taking over the active role so that IP redundancy clients can reply (either with an ok or wait reply) for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). Use the **no** form of the command to restore the default values.

	Command or Action	Purpose
Step 8	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 2 ip 10.0.0.4`	Creates the HSRP group using its number and virtual IP address. • (Optional) *group-number*- The group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) *ip-address*- The virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**- The IP address is a secondary hot standby router interface. If neither router is designated as a secondary or standby router and no priorities are set, the primary IP addresses are compared and the higher IP address is the active router, with the next highest as the standby router.
Step 9	**standby** [*group-number*] **preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:** `Device(config-if)# standby 2 preempt delay 300`	Configures the router to **preempt**, which means that when the local router has a higher priority than the active router, it becomes the active router. • (Optional) group-number-The group number to which the command applies. • (Optional) **delay minimum**—Set to cause the local router to postpone taking over the active role for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). • (Optional) **delay reload**—Set to cause the local router to postpone taking over the active role after a reload for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over after a reload). • (Optional) **delay sync**—Set to cause the local router to postpone taking over the active role so that IP redundancy clients

	Command or Action	Purpose
		can reply (either with an ok or wait reply) for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). Use the **no** form of the command to restore the default values.
Step 10	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 11	**show running-config**	Verifies the configuration of the standby groups.
Step 12	**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Configuring Router B

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet1/1`	Configures an interface type and enters interface configuration mode.
Step 3	**no switchport** **Example:** `Device(config)# no switchport`	Switches an interface that is in Layer 2 mode into Layer 3 mode for Layer 3 configuration.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.0.0.2 255.255.255.0`	Specifies an IP address for an interface.

	Command or Action	Purpose
Step 5	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 1 ip 10.0.0.3`	Creates the HSRP group using its number and virtual IP address. • (Optional) *group-number*- The group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) *ip-address*- The virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**- The IP address is a secondary hot standby router interface. If neither router is designated as a secondary or standby router and no priorities are set, the primary IP addresses are compared and the higher IP address is the active router, with the next highest as the standby router.
Step 6	**standby** [*group-number*] **priority** *priority* **Example:** `Device(config-if)# standby 2 priority 110`	Sets a **priority** value used in choosing the active router. The range is 1 to 255; the default priority is 100. The highest number represents the highest priority. (Optional) *group-number*—The group number to which the command applies. Use the **no** form of the command to restore the default values.
Step 7	**standby** [*group-number*] **preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:** `Device(config-if)# standby 1 preempt delay 300`	Configures the router to **preempt**, which means that when the local router has a higher priority than the active router, it becomes the active router. • (Optional) group-number-The group number to which the command applies. • (Optional) **delay minimum**—Set to cause the local router to postpone taking over the active role for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over).

	Command or Action	Purpose
		• (Optional) **delay reload**—Set to cause the local router to postpone taking over the active role after a reload for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over after a reload). • (Optional) **delay sync**—Set to cause the local router to postpone taking over the active role so that IP redundancy clients can reply (either with an ok or wait reply) for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). Use the **no** form of the command to restore the default values.
Step 8	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 2 ip 10.0.0.4`	Creates the HSRP group using its number and virtual IP address. • (Optional) *group-number*- The group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) *ip-address*- The virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**- The IP address is a secondary hot standby router interface. If neither router is designated as a secondary or standby router and no priorities are set, the primary IP addresses are compared and the higher IP address is the active router, with the next highest as the standby router.
Step 9	**standby** [*group-number*] **preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:**	Configures the router to **preempt**, which means that when the local router has a higher priority than the active router, it becomes the active router. • (Optional) group-number-The group number to which the command applies.

	Command or Action	Purpose
	`Device(config-if)# standby 2 preempt delay 300`	• (Optional) **delay minimum**—Set to cause the local router to postpone taking over the active role for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over) . • (Optional) **delay reload**—Set to cause the local router to postpone taking over the active role after a reload for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over after a reload). • (Optional) **delay sync**—Set to cause the local router to postpone taking over the active role so that IP redundancy clients can reply (either with an ok or wait reply) for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). Use the **no** form of the command to restore the default values.
Step 10	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 11	**show running-config**	Verifies the configuration of the standby groups.
Step 12	**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Configuring HSRP Authentication and Timers

You can optionally configure an HSRP authentication string or change the hello-time interval and hold-time interval.

When configuring these attributes, follow these guidelines:

- The authentication string is sent unencrypted in all HSRP messages. You must configure the same authentication string on all routers and access servers on a cable to ensure interoperation. Authentication mismatch prevents a device from learning the designated Hot Standby IP address and timer values from other routers configured with HSRP.

- Routers or access servers on which standby timer values are not configured can learn timer values from the active or standby router. The timers configured on an active router always override any other timer settings.
- All routers in a Hot Standby group should use the same timer values. Normally, the *holdtime* is greater than or equal to 3 times the *hellotime*.

Beginning in privileged EXEC mode, use one or more of these steps to configure HSRP authentication and timers on an interface:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config) # interface gigabitethernet1/1`	Enters interface configuration mode, and enter the HSRP interface on which you want to set priority.
Step 3	**standby** [*group-number*] **authentication** *string* **Example:** `Device(config-if) # standby 1 authentication word`	(Optional) **authentication** *string*—Enter a string to be carried in all HSRP messages. The authentication string can be up to eight characters in length; the default string is **cisco**. (Optional) *group-number*—The group number to which the command applies.
Step 4	**standby** [*group-number*] **timers** *hellotime holdtime* **Example:** `Device(config-if) # standby 1 timers 5 15`	(Optional) Configure the time interval to send and receive hello packets. • *group-number*—The group number to which the command applies. • *hellotime* —Set the interval between successive hello packets in seconds. The range is 1 to 255 seconds. The default is 3. • *holdtime*—Set the interval to wait for a hello packet from a neighbor device before declaring the neighbor device as inactive. The range is 1 to 255 seconds. The default is 10.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	
Step 6	**show running-config**	Verifies the configuration of the standby groups.
Step 7	**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Enabling HSRP Support for ICMP Redirect Messages

ICMP redirect messages are automatically enabled on interfaces configured with HSRP. ICMP is a network layer Internet protocol that provides message packets to report errors and other information relevant to IP processing. ICMP provides diagnostic functions, such as sending and directing error packets to the host. This feature filters outgoing ICMP redirect messages through HSRP, in which the next hop IP address might be changed to an HSRP virtual IP address. For more information, see the Cisco IOS IP Configuration Guide, Release 12.4.

Configuring HSRP Groups and Clustering

When a device is participating in an HSRP standby routing and clustering is enabled, you can use the same standby group for command switch redundancy and HSRP redundancy. Use the **cluster standby-group** *HSRP-group-name* [**routing-redundancy**] global configuration command to enable the same HSRP standby group to be used for command switch and routing redundancy. If you create a cluster with the same HSRP standby group name without entering the **routing-redundancy** keyword, HSRP standby routing is disabled for the group.

Verifying HSRP Configurations

From privileged EXEC mode, use this command to display HSRP settings:

show standby [*interface-id* [*group*]] [**brief**] [**detail**]

You can display HSRP information for the whole switch, for a specific interface, for an HSRP group, or for an HSRP group on an interface. You can also specify whether to display a concise overview of HSRP information or detailed HSRP information. The default display is **detail**. If there are a large number of HSRP groups, using the **show standby** command without qualifiers can result in an unwieldy display.

Example

```
Switch #show standby
VLAN1 - Group 1
Local state is Standby, priority 105, may preempt
Hellotime 3 holdtime 10
Next hello sent in 00:00:02.182
Hot standby IP address is 172.20.128.3 configured
Active router is 172.20.128.1 expires in 00:00:09
Standby router is local
Standby virtual mac address is 0000.0c07.ac01
Name is bbb

VLAN1 - Group 100
```

```
Local state is Standby, priority 105, may preempt
Hellotime 3 holdtime 10
Next hello sent in 00:00:02.262
Hot standby IP address is 172.20.138.51 configured
Active router is 172.20.128.1 expires in 00:00:09
Active router is local
Standby router is unknown expired
Standby virtual mac address is 0000.0c07.ac64
Name is test
```

Configuration Examples for Hot Standby Router Protocol

The following sections provide various configuration examples for HSRP.

Enabling HSRP: Example

This example shows how to activate HSRP for group 1 on an interface. The IP address used by the hot standby group is learned by using HSRP.

Note This procedure is the minimum number of steps required to enable HSRP. Other configurations are optional.

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/1
Switch(config-if)# no switchport
Switch(config-if)# standby 1 ip
Switch(config-if)# end
Switch # show standby
```

Example: Configuration and Verification for an HSRP Group

The following example shows configuration and verification for an HSRP group for IPv6 that consists of Device1 and Device2. The **show standby** command is issued for each device to verify the device's configuration:

Device 1 configuration

```
interface GigabitEthernet1/1
description DATA VLAN for PCs
encapsulation dot1Q 100
ipv6 address 2001:DB8:CAFE:2100::BAD1:1010/64
standby version 2
standby 101 priority 120
standby 101 preempt delay minimum 30
standby 101 authentication ese
standby 101 track Serial0/1/0.17 90
standby 201 ipv6 autoconfig
standby 201 priority 120
standby 201 preempt delay minimum 30
standby 201 authentication ese
standby 201 track Serial0/1/0.17 90
Device1# show standby
GigabitEthernet1/1 - Group 101 (version 2)
```

```
State is Active
2 state changes, last state change 5w5d
Active virtual MAC address is 0000.0c9f.f065
Local virtual MAC address is 0000.0c9f.f065 (v2 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 2.296 secs
Authentication text "ese"
Preemption enabled, delay min 30 secs
Active router is local
Priority 120 (configured 120)
Track interface Serial0/1/0.17 state Up decrement 90
GigabitEthernet1/1 - Group 201 (version 2)
State is Active
2 state changes, last state change 5w5d
Virtual IP address is FE80::5:73FF:FEA0:C9
Active virtual MAC address is 0005.73a0.00c9
Local virtual MAC address is 0005.73a0.00c9 (v2 IPv6 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 2.428 secs
Authentication text "ese"
Preemption enabled, delay min 30 secs
Active router is local
Standby router is FE80::20F:8FFF:FE37:3B70, priority 100 (expires in 7.856 sec)
Priority 120 (configured 120)
Track interface Serial0/1/0.17 state Up decrement 90
```

Device 2 configuration

```
interface GigabitEthernet1/2
description DATA VLAN for Computers
encapsulation dot1Q 100
ipv6 address 2001:DB8:CAFE:2100::BAD1:1020/64
standby version 2
standby 101 preempt
standby 101 authentication ese
standby 201 ipv6 autoconfig
standby 201 preempt
standby 201 authentication ese
Device2# show standby
GigabitEthernet1/2 - Group 101 (version 2)
State is Standby
7 state changes, last state change 5w5d
Active virtual MAC address is 0000.0c9f.f065
Local virtual MAC address is 0000.0c9f.f065 (v2 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 0.936 secs
Authentication text "ese"
Preemption enabled
MAC address is 0012.7fc6.8f0c
Standby router is local
Priority 100 (default 100)
GigabitEthernet1/2 - Group 201 (version 2)
State is Standby
7 state changes, last state change 5w5d
Virtual IP address is FE80::5:73FF:FEA0:C9
Active virtual MAC address is 0005.73a0.00c9
Local virtual MAC address is 0005.73a0.00c9 (v2 IPv6 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 0.936 secs
Authentication text "ese"
Preemption enabled
Active router is FE80::212:7FFF:FEC6:8F0C, priority 120 (expires in 7.548 sec)
MAC address is 0012.7fc6.8f0c
Standby router is local
Priority 100 (default 100)
```

Configuring HSRP Priority: Example

This example activates a port, sets an IP address and a priority of 120 (higher than the default value), and waits for 300 seconds (5 minutes) before attempting to become the active router:

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/1
Switch(config-if)# no switchport
Switch(config-if)# standby ip 172.20.128.3
Switch(config-if)# standby priority 120 preempt delay 300
Switch(config-if)# end
Switch # show standby
```

Configuring MHSRP: Example

This example shows how to enable the MHSRP configuration shown in the figure *MHSRP Load Sharing*

Router A Configuration

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 10.0.0.1 255.255.255.0
Switch(config-if)# standby ip 10.0.0.3
Switch(config-if)# standby 1 priority 110
Switch(config-if)#  standby 1 preempt
Switch(config-if)# standby 2 ip 10.0.0.4
Switch(config-if)#  standby 2 preempt
Switch(config-if)# end
```

Router B Configuration

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 10.0.0.2 255.255.255.0
Switch(config-if)# standby ip 10.0.0.3
Switch(config-if)#  standby 1 preempt
Switch(config-if)# standby 2 ip 10.0.0.4
Switch(config-if)# standby 2 priority 110
Switch(config-if)#  standby 2 preempt
Switch(config-if)# end
```

Configuring HSRP Authentication and Timer: Example

This example shows how to configure word as the authentication string required to allow Hot Standby routers in group 1 to interoperate:

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/1
Switch(config-if)# no switchport
Switch(config-if)# standby 1 authentication word
Switch(config-if)# end
```

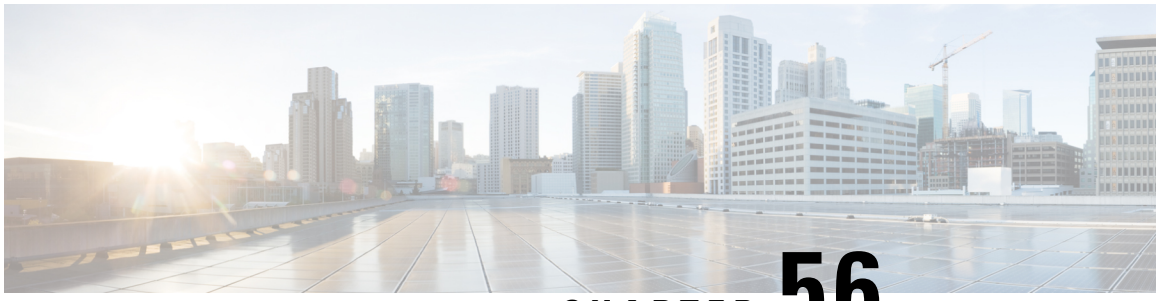
This example shows how to set the timers on standby group 1 with the time between hello packets at 5 seconds and the time after which a router is considered down to be 15 seconds:

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/1
Switch(config-if)# no switchport
Switch(config-if)# standby 1 ip
Switch(config-if)# standby 1 timers 5 15
Switch(config-if)# end
```

Configuring HSRP Groups and Clustering: Example

This example shows how to bind standby group my_hsrp to the cluster and enable the same HSRP group to be used for command switch redundancy and router redundancy. The command can only be executed on the cluster command switch. If the standby group name or number does not exist, or if the switch is a cluster member switch, an error message appears.

```
Switch # configure terminal
Switch(config) # cluster standby-group my_hsrp routing-redundancy
Switch(config-if)# end
```



CHAPTER 56

VRRPv3 Protocol Support

Restrictions for VRRPv3 Protocol Support

- VRRPv3 is not intended as a replacement for existing dynamic protocols. VRRPv3 is designed for use over multi-access, multicast, or broadcast capable Ethernet LANs.
- Because of the forwarding delay that is associated with the initialization of a BVI interface, you must not configure the VRRPv3 advertise timer to a value lesser than the forwarding delay on the BVI interface. If you configure the VRRPv3 advertise timer to a value equal to or greater than the forwarding delay on the BVI interface, the setting prevents a VRRP device on a recently initialized BVI interface from unconditionally taking over the primary role. Use the **bridge forward-time** command to set the forwarding delay on the BVI interface. Use the **vrrp timers advertise** command to set the VRRP advertisement timer.
- Full network redundancy can only be achieved if VRRP operates over the same network path as the VRRS Pathway redundant interfaces. For full redundancy, the following restrictions apply:
  - VRRS pathways should not share a different physical interface as the parent VRRP group or be configured on a sub-interface having a different physical interface as the parent VRRP group.
  - VRRS pathways should not be configured on Switch Virtual Interface (SVI) interfaces as long as the associated VLAN does not share the same trunk as the VLAN on which the parent VRRP group is configured.
- Interface link-local IP address and VRRP group virtual link-local IP address should be different for VRRP features to work properly.

Information About VRRPv3 Protocol Support

The following sections provide information about VRRPv3 protocol support.

VRRPv3 Benefits

Support for IPv4 and IPv6

VRRPv3 supports IPv4 and IPv6 address families while VRRPv2 only supports IPv4 addresses.

Note When VRRPv3 is in use, VRRPv2 is unavailable. For VRRPv3 to be configurable, the **fhrp version vrrp v3** command must be used in global configuration mode

Redundancy

VRRP enables you to configure multiple devices as the default gateway device, which reduces the possibility of a single point of failure in a network.

Load Sharing

You can configure VRRP in such a way that traffic to and from LAN clients can be shared by multiple devices, thereby sharing the traffic load more equitably between available devices.

Multiple Virtual Devices

VRRP supports up to 255 virtual devices (VRRP groups) on a device physical interface, subject to restrictions in scaling. Multiple virtual device support enables you to implement redundancy and load sharing in your LAN topology. In scaled environments, VRRS Pathways should be used in combination with VRRP control groups.

Multiple IP Addresses

The virtual device can manage multiple IP addresses, including secondary IP addresses. Therefore, if you have multiple subnets configured on an Ethernet interface, you can configure VRRP on each subnet.

Note To utilize secondary IP addresses in a VRRP group, a primary address must be configured on the same group.

Preemption

The redundancy scheme of VRRP enables you to preempt a virtual device backup that has taken over for a failing primary virtual device with a higher priority virtual device backup that has become available.

Note Preemption of a lower priority primary device is enabled with an optional delay.

Advertisement Protocol

VRRP uses a dedicated Internet Assigned Numbers Authority (IANA) standard multicast address for VRRP advertisements. For IPv4, the multicast address is 224.0.0.18. For IPv6, the multicast address is FF02:0:0:0:0:0:0:12. This addressing scheme minimizes the number of devices that must service the multicasts

and allows test equipment to accurately identify VRRP packets on a segment. The IANA has assigned VRRP the IP protocol number 112.

VRRP Device Priority and Preemption

An important aspect of the VRRP redundancy scheme is VRRP device priority. Priority determines the role that each VRRP device plays and what happens if the primary virtual device fails.

If a VRRP device owns the IP address of the virtual device and the IP address of the physical interface, this device will function as a primary virtual device.

Priority also determines if a VRRP device functions as a virtual device backup and the order of ascendancy to becoming a primary virtual device if the primary virtual device fails. You can configure the priority of each virtual device backup with a value of 1 through 254 using the **priority** command (use the **vrrp address-family** command to enter the VRRP configuration mode and access the **priority** option).

For example, if device A, the primary virtual device in a LAN topology, fails, an election process takes place to determine if virtual device backups B or C should take over. If devices B and C are configured with the priorities of 101 and 100, respectively, device B is elected to become primary virtual device because it has the higher priority. If devices B and C are both configured with the priority of 100, the virtual device backup with the higher IP address is elected to become the primary virtual device.

By default, a preemptive scheme is enabled whereby a higher priority virtual device backup that becomes available takes over from the virtual device backup that was elected to become primary virtual device. You can disable this preemptive scheme using the **no preempt** command (use the **vrrp address-family** command to enter the VRRP configuration mode, and enter the **no preempt** command). If preemption is disabled, the virtual device backup that is elected to become primary virtual device remains the primary until the original primary virtual device recovers and becomes primary again.

Note Preemption of a lower priority primary device is enabled with an optional delay.

VRRP Advertisements

The primary virtual device sends VRRP advertisements to other VRRP devices in the same group. The advertisements communicate the priority and state of the primary virtual device. The VRRP advertisements are encapsulated into either IPv4 or IPv6 packets (based on the VRRP group configuration) and sent to the appropriate multicast address assigned to the VRRP group. For IPv4, the multicast address is 224.0.0.18. For IPv6, the multicast address is FF02:0:0:0:0:0:0:12. The advertisements are sent every second by default and the interval is configurable.

Cisco devices allow you to configure millisecond timers, which is a change from VRRPv2. You need to manually configure the millisecond timer values on both the primary and the backup devices. The primary advertisement value displayed in the **show vrrp** command output on the backup devices is always 1 second because the packets on the backup devices do not accept millisecond values.

You must use millisecond timers where absolutely necessary and with careful consideration and testing. Millisecond values work only under favorable circumstances. The use of the millisecond timer values is compatible with third party vendors, as long as they also support VRRPv3. You can specify a timer value between 100 milliseconds and 40000 milliseconds.

How to Configure VRRPv3 Protocol Support

The following sections provide configuration information about VRRPv3 protocol support.

Creating and Customizing a VRRP Group

To create a VRRP group, perform the following task. Steps 6 to 14 denote customizing options for the group, and they are optional:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**fhrp version vrrp v3** **Example:** `Device(config)# fhrp version vrrp v3`	Enables the ability to configure VRRPv3 and VRRS.
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Enters interface configuration mode.
Step 5	**vrrp** *group-id* **address-family** {**ipv4** | **ipv6**} **Example:** `Device(config-if)# vrrp 3 address-family ipv4`	Creates a VRRP group and enters VRRP configuration mode.
Step 6	**address** *ip-address* [**primary** | **secondary**] **Example:** `Device(config-if-vrrp)# address 100.0.1.10 primary`	Specifies a primary or secondary address for the VRRP group. **Note** VRRPv3 for IPv6 requires that a primary virtual link-local IPv6 address is configured to allow the group to operate. After the primary link-local IPv6 address is established

	Command or Action	Purpose
		on the group, you can add the secondary global addresses.
Step 7	**description** *group-description* **Example:** `Device(config-if-vrrp)# description group 3`	(Optional) Specifies a description for the VRRP group.
Step 8	**match-address** **Example:** `Device(config-if-vrrp)# match-address`	(Optional) Matches secondary address in the advertisement packet against the configured address. **Note** Secondary address matching is enabled by default.
Step 9	**preempt delay minimum** *seconds* **Example:** `Device(config-if-vrrp)# preempt delay minimum 30`	(Optional) Enables preemption of lower priority primary device with an optional delay. **Note** Preemption is enabled by default.
Step 10	**priority** *priority-level* **Example:** `Device(config-if-vrrp)# priority 3`	(Optional) Specifies the priority value of the VRRP group. The priority of a VRRP group is 100 by default.
Step 11	**timers advertise** *interval* **Example:** `Device(config-if-vrrp)# timers advertise 1000`	(Optional) Sets the advertisement timer in milliseconds. The advertisement timer is set to 1000 milliseconds by default.
Step 12	**vrrpv2** **Example:** `Device(config-if-vrrp)# vrrpv2`	(Optional) Enables support for VRRPv2 configured devices in compatibility mode.
Step 13	**vrrs leader** *vrrs-leader-name* **Example:** `Device(config-if-vrrp)# vrrs leader leader-1`	(Optional) Specifies a leader's name to be registered with VRRS and to be used by followers. **Note** A registered VRRS name is unavailable by default.
Step 14	**shutdown** **Example:**	(Optional) Disables VRRP configuration for the VRRP group. **Note**

	Command or Action	Purpose
	`Device(config-if-vrrp)# shutdown`	VRRP configuration is enabled for a VRRP group by default.
Step 15	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Delay Period Before FHRP Client Initialization

To configure the delay period before the initialization of all FHRP clients on an interface, perform the following task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**fhrp version vrrp v3** **Example:** `Device(config)# fhrp version vrrp v3`	Enables the ability to configure VRRPv3 and VRRS.
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Enters interface configuration mode.
Step 5	**fhrp delay** {[**minimum**] [**reload**] *seconds*} **Example:** `Device(config-if)# fhrp delay minimum 5`	Specifies the delay period for the initialization of FHRP clients after an interface comes up. The range is 0-3600 seconds.
Step 6	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	

Configuration Examples for VRRPv3 Protocol Support

The following sections provide configuration examples for VRRPv3 protocol support.

Example: Enabling VRRPv3 on a Device

The following example shows how to enable VRRPv3 on a device:

```
Device> enable
Device# configure terminal
Device(config)# fhrp version vrrp v3
Device(config-if-vrrp)# end
```

Example: Creating and Customizing a VRRP Group

The following example shows how to create and customize a VRRP group:

```
Device> enable
Device# configure terminal
Device(config)# fhrp version vrrp v3
Device(config)# interface GigabitEthernet 1/1
Device(config-if)# vrrp 3 address-family ipv4
Device(config-if-vrrp)# address 100.0.1.10 primary
Device(config-if-vrrp)# description group 3
Device(config-if-vrrp)# match-address
Device(config-if-vrrp)# preempt delay minimum 30
Device(config-if-vrrp)# end
```

Note In the above example, the **fhrp version vrrp v3** command is used in the global configuration mode.

Example: Configuring the Delay Period Before FHRP Client Initialization

The following example shows how to configure the delay period before FHRP client initialization :

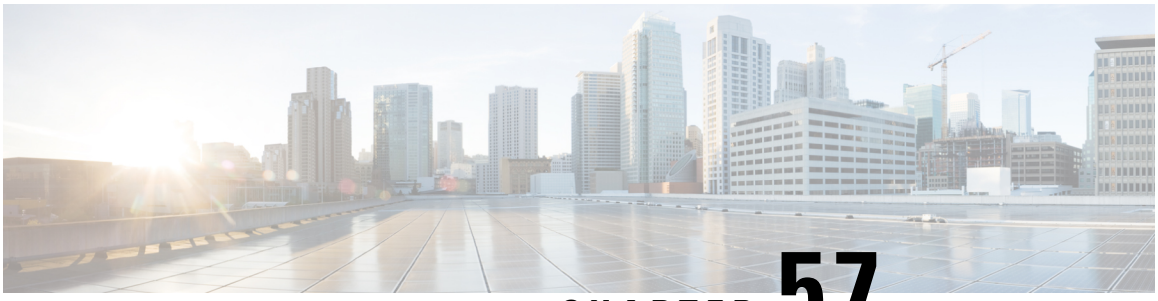
```
Device> enable
Device# configure terminal
Device(config)# fhrp version vrrp v3
Device(config)# interface GigabitEthernet 1/1
Device(config-if)# fhrp delay minimum 5
Device(config-if-vrrp)# end
```

Note In the above example, a five-second delay period is specified for the initialization of FHRP clients after the interface comes up. You can specify a delay period between 0 and 3600 seconds.

Example: VRRP Status, Configuration, and Statistics Details

The following is a sample output of the status, configuration, and statistics details for a VRRP group:

```
Device> enable
Device# show vrrp detail

 GigabitEthernet1/1 - Group 3 - Address-Family IPv4
  Description is "group 3"
  State is MASTER
  State duration 53.901 secs
  Virtual IP address is 100.0.1.10
  Virtual MAC address is 0000.5E00.0103
  Advertisement interval is 1000 msec
  Preemption enabled, delay min 30 secs (0 msec remaining)
  Priority is 100
  Master Router is 10.21.0.1 (local), priority is 100
  Master Advertisement interval is 1000 msec (expires in 832 msec)
  Master Down interval is unknown
  VRRPv3 Advertisements: sent 61 (errors 0) - rcvd 0
  VRRPv2 Advertisements: sent 0 (errors 0) - rcvd 0
  Group Discarded Packets: 0
    VRRPv2 incompatibility: 0
    IP Address Owner conflicts: 0
    Invalid address count: 0
    IP address configuration mismatch : 0
    Invalid Advert Interval: 0
    Adverts received in Init state: 0
    Invalid group other reason: 0
  Group State transition:
    Init to master: 0
    Init to backup: 1 (Last change Sun Mar 13 19:52:56.874)
    Backup to master: 1 (Last change Sun Mar 13 19:53:00.484)
    Master to backup: 0
    Master to init: 0
    Backup to init: 0

Device# exit
```



CHAPTER 57

Configuring Enhanced Object Tracking

Information About Enhanced Object Tracking

The following sections provide information about enhanced object tracking.

Enhanced Object Tracking Overview

Before the introduction of the Enhanced Object Tracking feature, Hot Standby Router Protocol (HSRP) had a simple tracking mechanism that allowed you to track the interface line-protocol state only. If the line-protocol state of the interface went down, the HSRP priority of the router was reduced, allowing another HSRP router with a higher priority to become active.

The Enhanced Object Tracking feature separates the tracking mechanism from HSRP and creates a separate standalone tracking process that can be used by processes other than HSRP. This feature allows the tracking of other objects in addition to the interface line-protocol state.

A client process such as HSRP, Virtual Router Redundancy Protocol (VRRP), or Gateway Load Balancing Protocol (GLBP), can register its interest in tracking objects and then be notified when the tracked object changes state.

Each tracked object has a unique number that is specified in the tracking command-line interface (CLI). Client processes use this number to track a specific object. The tracking process periodically polls the tracked object for value changes and sends any changes (as up or down values) to interested client processes, either immediately or after a specified delay. Several clients can track the same object, and can take different actions when the object changes state.

You can also track a combination of objects in a list by using either a weight threshold or a percentage threshold to measure the state of the list. You can combine objects using Boolean logic. A tracked list with a Boolean “AND” function requires that each object in the list be in an up state for the tracked object to be up. A tracked list with a Boolean “OR” function needs only one object in the list to be in the up state for the tracked object to be up.

Tracking Interface Line-Protocol or IP Routing State

You can track either the interface line protocol state or the interface IP routing state. When you track the IP routing state, these three conditions are required for the object to be up:

- IP routing must be enabled and active on the interface.
- The interface line-protocol state must be up.
- The interface IP address must be known.

If all three of these conditions are not met, the IP routing state is down.

Tracked Lists

You can configure a tracked list of objects with a Boolean expression, a weight threshold, or a percentage threshold. A tracked list contains one or more objects. An object must exist before it can be added to the tracked list.

- You configure a Boolean expression to specify calculation by using either “AND” or “OR” operators.
- When you measure the tracked list state by a weight threshold, you assign a weight number to each object in the tracked list. The state of the tracked list is determined by whether or not the threshold was met. The state of each object is determined by comparing the total weight of all objects against a threshold weight for each object.
- When you measure the tracked list by a percentage threshold, you assign a percentage threshold to all objects in the tracked list. The state of each object is determined by comparing the assigned percentages of each object to the list.

Tracking Other Characteristics

You can also use the enhanced object tracking for tracking other characteristics.

- You can track the reachability of an IP route by using the **track ip route reachability** global configuration command.
- You can use the **track ip route metric threshold** global configuration command to determine if a route is above or below threshold.
- You can use the **track resolution** global configuration command to change the metric resolution default values for routing protocols.
- You can use the **track timer tracking** configuration command to configure the tracking process to periodically poll tracked objects.

Use the **show track** privileged EXEC command to verify enhanced object tracking configuration.

IP SLAs Object Tracking

Cisco IOS IP Service Level Agreements (IP SLAs) is a network performance measurement and diagnostics tool that uses active monitoring by generating traffic to measure network performance. Cisco IP SLAs operations collects real-time metrics that you can use for network troubleshooting, design, and analysis.

Object tracking of IP SLAs operations allows clients to track the output from IP SLAs objects and use this information to trigger an action. Every IP SLAs operation maintains an SNMP operation return-code value, such as OK or OverThreshold, that can be interpreted by the tracking process. You can track two aspects of IP SLAs operation: state and reachability. For state, if the return code is OK, the track state is up; if the return code is not OK, the track state is down. For reachability, if the return code is OK or OverThreshold, reachability is up; if not OK, reachability is down.

Static Route Object Tracking

Static routing support using enhanced object tracking provides the ability for the device to use ICMP pings to identify when a pre-configured static route or a DHCP route goes down. When tracking is enabled, the system tracks the state of the route and informs the client when that state changes. Static route object tracking uses Cisco IP SLAs to generate ICMP pings to monitor the state of the connection to the primary gateway.

How to Configure Enhanced Object Tracking

The following sections provide configuration information about enhanced object tracking.

Configuring Tracking for Line State Protocol or IP Routing State on an Interface

Follow these steps to track the line-protocol state or IP routing state of an interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *object-number* **interface** *interface-id* **line-protocol** **Example:** `Device(config)# track 33 interface gigabitethernet 1/1 line-protocol`	(Optional) Creates a tracking list to track the line-protocol state of an interface and enter tracking configuration mode. • The object-number identifies the tracked object and can be from 1 to 500. • The**interface** interface-id is the interface being tracked.

	Command or Action	Purpose
Step 4	**delay** { *object-number* **up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time in seconds to delay communicating state changes of a tracked object. The range is from 1 to 180 seconds.
Step 5	**exit**	Returns to global configuration mode.
Step 6	**track** *object-number* **interface** *interface-id* **ip routing** **Example:** `Device(config)# track 33 interface gigabitethernet 1/1 ip routing`	(Optional) Creates a tracking list to track the IP routing state of an interface and enter tracking configuration mode. IP route tracking tracks an IP route in the routing table and the ability of an interface to route IP packets. • The object-number identifies the tracked object and can be from 1 to 500. • The**interface** interface-id is the interface being tracked.
Step 7	**delay** { *object-number* **up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time in seconds to delay communicating state changes of a tracked object. The range is from 1 to 180 seconds.
Step 8	**end**	Returns to privileged EXEC mode.
Step 9	**show track***object-number*	Verifies that the specified objects are being tracked.

Configuring Tracked Lists

The following sections provide configuration information about tracked lists.

Configuring a Tracked List with a Weight Threshold

To track by weight threshold, configure a tracked list of objects, specify that weight is used as the threshold, and configure a weight for each of its objects. The state of each object is determined by comparing the total weight of all objects that are up against a threshold weight for each object.

You cannot use the Boolean "NOT" operator in a weight threshold list.

Follow these steps to configure a tracked list of objects by using a weight threshold and to configure a weight for each object:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *track-number* **list threshold** {**weight**} **Example:** `Device(config)# track 4 list threshold weight`	Configures a tracked list object, and enters tracking configuration mode. The track-number can be from 1 to 500. • **threshold—**Specifies the state of the tracked list based on a threshold. • **weight—** Specifies that the threshold is based on weight.
Step 4	**object** *object-number* [**weight** *weight-number*] **Example:** `Device(config)#object 2 weight 15`	Specifies the object to be tracked. The range is from 1 to 500. The optional**weight***weight-number* specifies the threshold weight for the object. The range is from 1 to 255. **Note** An object must exist before you can add it to a tracked list.
Step 5	**threshold weight** {**up** *number* | [**down** *number*]} **Example:** `Device(config-track)# threshold weight up 30 down 10`	(Optional) Specifies the threshold weight. • **up***number*—The range is from 1 to 255. • **down***number*—(Optional)The range depends on the number selected for the **up***number*. If you configure the **up***number* as 25, the range shown for the down number is 0 to 24.
Step 6	**delay** { **up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time in seconds to delay communicating state changes of a tracked object. The range is from 1 to 180 seconds.
Step 7	**end**	Returns to privileged EXEC mode.
Step 8	**show track** *object-number*	Verify that the specified objects are being tracked.
Step 9	**copy running-config startup-config** **Example:**	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`Device# copy running-config startup-config`	

Configuring a Tracked List with a Percentage Threshold

To track by percentage threshold, configure a tracked list of objects, specify that a percentage will be used as the threshold, and specify a percentage for all objects in the list. The state of the list is determined by comparing the assigned percentage of each object to the list.

You cannot use the Boolean "NOT" operator in a percentage threshold list.

Follow these steps to configure a tracked list of objects by using a percentage threshold:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *track-number* **list threshold** {**percentage**} **Example:** `Device(config)# track 4 list threshold percentage`	Configures a tracked list object, and enters tracking configuration mode. The track-number can be from 1 to 500. • **threshold—**Specifies the state of the tracked list based on a threshold. • **percentage—** Specifies that the threshold is based on percentage.
Step 4	**object** *object-number* **Example:** `Device(config)# object 1`	Specifies the object to be tracked. The range is from 1 to 500. **Note** An object must exist before you can add it to a tracked list.
Step 5	**threshold percentage** {**up** *number* | [**down***number*]} **Example:**	(Optional) Specifies the threshold percentage. • **up***number*—The range is from 1 to 100.

	Command or Action	Purpose
	`Device(config)# threshold percentage up 51 down 10`	• **down***number*—(Optional)The range depends on the number selected for the **up***number*. If you configure the **up***number* as 25, the range shown for the down number is 0 to 24.
Step 6	**delay** { **up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time in seconds to delay communicating state changes of a tracked object. The range is from 1 to 180 seconds.
Step 7	**end**	Returns to privileged EXEC mode.
Step 8	**show track** *object-number*	Verify that the specified objects are being tracked.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring HSRP Object Tracking

Follow these steps to configure a standby HSRP group to track an object and change the HSRP priority based on the object state:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *object-number* {**interface** *interface-id* {**line-protocol** | **ip routing**} | **ip route** *ip address/prefix-length* {**metric threshold** |	(Optional) Create a tracking list to track the configured state and enter tracking configuration mode.

	Command or Action	Purpose
	reachability} **list** {**boolean** {**and** | **or**}} | {**threshold** {**weight** | **percentage**}}}	• The object-number identifies the tracked object and can be from 1 to 500. • Enter**interface** *interface-id* to select an interface to track. • Enter**line-protocol** to track the interface line protocol state or enter **ip routing** to track the interface IP routing state . • Enter**ip route** *ip-address/prefix-length* to track the state of an IP route. • Enter**metric threshold** to track the threshold metric or enter **reachability** to track if the route is reachable. The default up threshold is 254 and the default down threshold is 255. • Enter **list** to track objects grouped in a list. **Note** Repeat this step for each interface to be tracked.
Step 4	**exit**	Return to global configuration mode.
Step 5	**interface** { *interface-id*	Enter interface configuration mode.
Step 6	**standby** [*group-number*] **ip** [*ip-address* **secondary**]]	Creates (or enables) the HSRP group by using its number and virtual IP address. • (Optional) *group-number*—Enters a group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) *ip-address*—Specifies the virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**—Specifies that the IP address is a secondary hot standby router interface. If this keyword is omitted, the configured address is the primary IP address.

	Command or Action	Purpose
Step 7	**standby** [*group-number*] **track** [*object-number* [**decrement** *priority-decrement*]]	Configures HSRP to track an object and change the hot standby priority based on the state of the object. • (Optional) *group-number*—Enters the group number to which the tracking applies. • *object-number*—Enters a number representing the object to be tracked. The range is from 1 to 500; the default is 1. • (Optional) **secondary**—Specifies that the IP address is a secondary hot standby router interface. If this keyword is omitted, the configured address is the primary IP address. • (Optional) **decrement** *priority-decrement*—Specifies the amount by which the hot standby priority for the router is decremented (or incremented) when the tracked object goes down (or comes back up). The range is from 1 to 255; the default is 10.
Step 8	**end**	Returns to privileged EXEC mode.
Step 9	**show standby**	Verifies the standby router IP address and tracking states.
Step 10	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring IP SLAs Object Tracking

Follow these steps to track the state of an IP SLAs operation or the reachability of an IP SLAs IP host:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *object-number* **ip sla** *operation-number* {**state** | **reachability**} **Example:** `Device(config)# track 2 ip sla 123 state`	Enters tracking configuration mode to track the state of an IP SLAs operation. • *object-number* range is from 1 to 500. • *operation-number* range is from 1 to 2147483647.
Step 4	**delay** { **up***seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time in seconds to delay communicating state changes of a tracked object. The range is from 1 to 180 seconds.
Step 5	**end**	Returns to privileged EXEC mode.
Step 6	**show track** *object-number*	Verifies that the specified objects are being tracked.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Static Route Object Tracking

The following sections provide configuration information about static route object tracking.

Configuring a Primary Interface for Static Routing

Follow these steps to configure a primary interface for static routing:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id*	Selects a primary or secondary interface and enters interface configuration mode.
Step 4	**description** *string*	Adds a description to the interface.
Step 5	**ip address** *ip-address mask* [**secondary**]	Sets the primary or secondary IP address for the interface.
Step 6	**exit**	Returns to global configuration mode.

Configuring a Primary Interface for DHCP

Follow these steps to configure a primary interface for DHCP:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id*	Selects a primary or secondary interface and enters interface configuration mode.
Step 4	**description** *string*	Adds a description to the interface.
Step 5	**ip dhcp client route track** *number*	Configures the DCHP client to associate any added routes with the specified track number. Valid numbers are from 1 to 500.
Step 6	**exit**	Returns to global configuration mode.

Configuring IP SLAs Monitoring Agent

You can configure an IP SLAs agent to ping an IP address using a primary interface and a track object to monitor the state of the agent.

Follow these steps to configure network monitoring with Cisco IP SLAs:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip sla** *operation number*	Begins configuring a Cisco IP SLAs operation and enters IP SLA configuration mode.
Step 4	**icmp-echo** { *destination ip-address* | *destination hostname* [**source - ipaddr** {*ip-address* | *hostname* **source-interface** *interface-id*]	Configures a Cisco IP SLAs end-to-end ICMP echo response time operation and enter IP SLAs ICMP echo configuration mode.
Step 5	**timeout** *milliseconds*	Sets the amount of time for which the operation waits for a response from its request packet.
Step 6	**frequency** *seconds*	Sets the rate at which the operation is sent into the network.
Step 7	**threshold** *milliseconds*	Sets the rising threshold (hysteresis) that generates a reaction event and stores history information for the operation.
Step 8	**exit**	Exits IP SLAs ICMP echo configuration mode.
Step 9	**ip sla schedule** *operation-number* [**life** {**forever** | *seconds*}] **start-time** *time* | **pending** | **now** | **after** *time*] **ageout** *seconds*] [**recurring**] **Example:** `Device(config)# track 2 200 state`	Configures the scheduling parameters for a single IP SLAs operation. • *object-number* range is from 1 to 500. • *operation-number* range is from 1 to 2147483647.

	Command or Action	Purpose
Step 10	**track** *object-number* **rtr** *operation-number* **state reachability**	Tracks the state of a Cisco IOS IP SLAs operation and enter tracking configuration mode.
Step 11	**end**	Returns to privileged EXEC mode.
Step 12	**show track** *object-number*	Verifies that the specified objects are being tracked.
Step 13	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Routing Policy and a Default Route

Follow these steps to configure a routing policy for backup static routing by using object tracking.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *access-list-number*	Defines an extended IP access list. Configure any optional characteristics.
Step 4	**route-map** *map tag* [**permit** | **deny**] [*sequence-number*]	Enters route-map configuration mode and define conditions for redistributing routes from one routing protocol to another.
Step 5	**match ip address** {*access-list number* [**permit**|**deny**] [*sequence-number*]	Distribute any routes that have a destination network number address that is permitted by a standard or extended access list or performs policy routing on packets. You can enter multiple numbers or names.

	Command or Action	Purpose
Step 6	**set ip next-hop dynamic dhcp**	For DHCP networks only. Sets the next hop to the gateway that was most recently learned by the DHCP client.
Step 7	**set interface** *interface-id*	For static routing networks only. Indicates where to send output packets that pass a match clause of a route map for policy routing.
Step 8	**exit**	Returns to global configuration mode.
Step 9	**ip local policy route-map** *map tag*	Identifies a route map to use for local policy routing.
Step 10	**ip route** *prefix mask* {*ip address* | *interface-id* [*ip address*]} [*distance*] [*name*] [**permanent** | **track** *track-number*] [*tag tag*]	For static routing networks only. Establishes static routes. Entering **track***track-number* specifies that the static route is installed only if the configured track object is up.
Step 11	**end**	Returns to privileged EXEC mode.
Step 12	**show ip route track table**	Displays information about the IP route track table.
Step 13	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

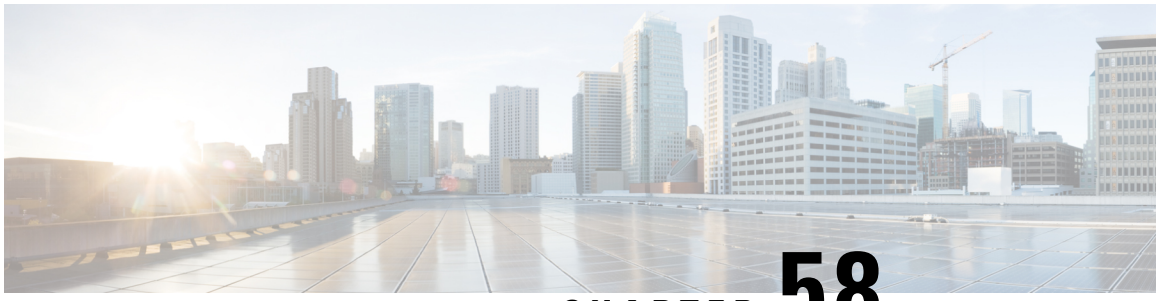
Monitoring Enhanced Object Tracking

Use the privileged EXEC or user EXEC commands in the table below, to display enhanced object tracking information.

Table 60: Commands for Displaying Tracking Information

Command	Purpose
show ip route track table	Displays information about the IP route track table.
show track [*object-number*]	Displays information about the all tracking lists or the specified list.
show track brief	Displays VTP status and configuration for all interfaces or the specified interface.
show track interface [brief]	Displays information about tracked interface objects.
show track ip [*object-number*] [**brief**] **route**	Displays information about tracked IP-route objects

Command	Purpose
show track resolution	Displays the resolution of tracked parameters.
show track timer	Displays tracked polling interval timers.



CHAPTER 58

Configuring TCP MSS Adjustment

Restrictions for TCP MSS Adjustment

- TCP MSS adjustment configuration works only if applied on an ingress interface. This configuration does not work if applied on an egress interface.

Information about TCP MSS Adjustment

The Transmission Control Protocol (TCP) Maximum Segment Size (MSS) Adjustment feature enables the configuration of the maximum segment size for transient packets that traverse a router, specifically TCP segments with the SYN bit set. Use the **ip tcp adjust-mss** command in interface configuration mode to specify the MSS value on the intermediate router of the SYN packets to avoid truncation.

When a host (usually a PC) initiates a TCP session with a server, it negotiates the IP segment size by using the MSS option field in the TCP SYN packet. The value of the MSS field is determined by the MTU configuration on the host. The default MSS value for a PC is 1500 bytes.

The **ip tcp adjust-mss** command helps prevent TCP sessions from being dropped by adjusting the MSS value of the TCP SYN packets.

The **ip tcp adjust-mss** command is effective only for TCP connections passing through the router.

In most cases, the optimum value for the *max-segment-size* argument of the **ip tcp adjust-mss** command is 1452 bytes.

Note TCP MSS adjustment-based traffic is always software switched.

Supported Interfaces

TCP MSS Adjust is supported only on the following interfaces:

- Physical Layer 3 interface
- SVI
- Layer 3 port channel
- Layer 3 GRE tunnel

How to Configure TCP MSS Adjustment

The following sections provide configuration information for TCP MSS adjustment.

Configuring the MSS Value for Transient TCP SYN Packets

Before you begin

Perform this task to configure the MSS for transient packets that traverse a router, specifically TCP segments with the SYN bit set.

We recommend that you use **ip tcp adjust-mss 1452** command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted
Step 2	**configure terminal** **Example:** `Device#config terminal`	Enters the global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)#interface Gigabitethernet 1/1`	Configures an interface type and enters interface configuration mode.
Step 4	**ip tcp adjust-mss** *max-segment-size* **Example:** `Device(config-if)#ip tcp adjust-mss 1452`	Adjusts the MSS value of TCP SYN packets going through a router. The max-segment-size argument is the maximum segment size, in bytes. The range is from 500 to 1460.
Step 5	**end** **Example:** `Device(config-if)#end`	Exits to global configuration mode.

Configuring the MSS Value for IPv6 Traffic

Procedure

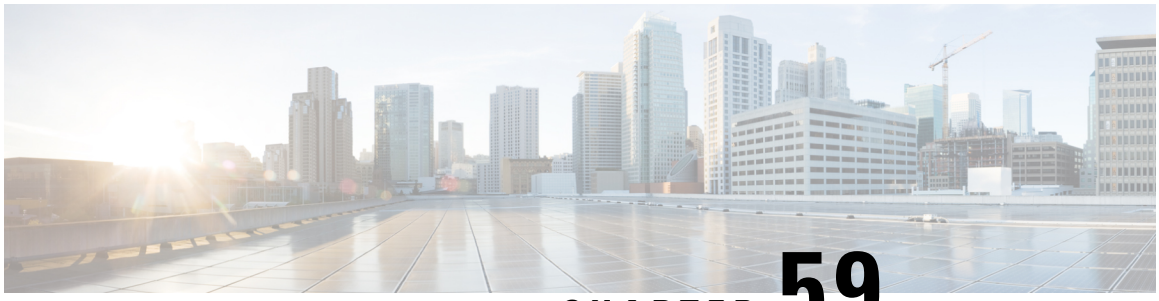
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted
Step 2	**configure terminal** **Example:** `Device#config terminal`	Enters the global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)#interface Gigabitethernet 1/1`	Configures an interface type and enters interface configuration mode.
Step 4	**ipv6 tcp adjust-mss** *max-segment-size* **Example:** `Device(config-if)#ipv6 tcp adjust-mss 1440`	Adjusts the MSS value of TCP DF packets going through a device. The max-segment-size argument is the maximum segment size, in bytes. The range is from 40 to 1440.
Step 5	**end** **Example:** `Device(config-if)#end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for TCP MSS Adjustment

The following sections provide configuration examples for TCP MSS adjustment.

Example: Configuring the TCP MSS Adjustment for IPv6 traffic

```
Device>enable
Device#configure terminal
Device(config)#interface GigabitEthernet 1/1
Device(config)#ipv6 tcp adjust-mss 1440
Device(config)#end
```



CHAPTER 59

Enhanced IPv6 Neighbor Discovery Cache Management

Enhanced IPv6 Neighbor Discovery Cache Management

Neighbor discovery protocol enforces the neighbor unreachability detection process to detect failing nodes, or devices, and the changes to link-layer addresses. Neighbor unreachability detection process maintains the reachability information for all the paths between hosts and neighboring nodes, including host-to-host, host-to-device, and device-to-host communication.

The neighbor cache maintains mapping information about the IPv6 link-local or global address to the link-layer address. The neighbor cache also maintains the reachability state of the neighbor using the neighbor unreachability detection process. Neighbors can be in one of the following five possible states:

- DELAY: Neighbor resolution is pending, and traffic might flow to this neighbor.
- INCOMPLETE: Address resolution is in progress, and the link-layer address is not yet known.
- PROBE: Neighbor resolution is in progress, and traffic might flow to this neighbor.
- REACHABLE: Neighbor is known to be reachable within the last reachable time interval.
- STALE: Neighbor requires resolution, and traffic may flow to this neighbor.

Use the **ipv6 nd na glean** command to configure the neighbor discovery protocol to glean an entry from an unsolicited neighbor advertisement.

Use the **ipv6 nd nud retry** command to configure the neighbor discovery protocol to maintain a neighbor discovery cache entry for a neighbor during a network disruption.

Use the **ipv6 nd cache expire refresh** command to configure the neighbor discovery protocol to maintain a neighbor discovery cache entry even when no traffic flows to the neighbor.

Customizing the Parameters for IPv6 Neighbor Discovery

To customize the parameters for IPv6 neighbor discovery, perform this procedure:

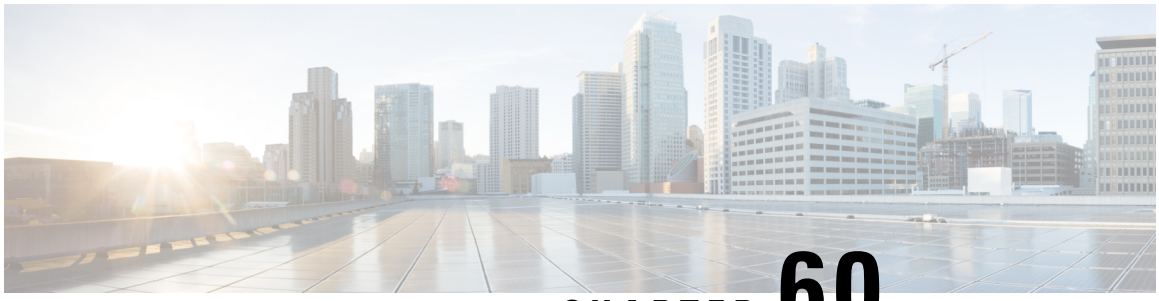
Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/4`	Specifies an interface type and identifier. Enters the interface configuration mode.
Step 4	**ipv6 nd nud retry** *base interval max-attempts* [*final-wait-time*] **Example:** `Device(config-if)# ipv6 nd nud retry 1 1000 3`	Configures the number of times neighbor unreachability detection resends neighbor solicitations.
Step 5	**ipv6 nd cache expire** *expire-time-in-seconds* [**refresh**] **Example:** `Device(config-if)# ipv6 nd cache expire 7200`	Configures the length of time before an IPv6 neighbor discovery cache entry expires.
Step 6	**ipv6 nd na glean** **Example:** `Device(config-if)# ipv6 nd na glean`	Configures the length of time before an IPv6 neighbor discovery cache entry expires.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 8	**show ipv6 interface** **Example:** `Device# show ipv6 interface`	(Optional) Displays the usability status of interfaces that are configured for IPv6 along with neighbor discovery cache management.

Examples: Customizing Parameters for IPv6 Neighbor Discovery

The following example shows that IPv6 neighbor advertisement gleaning is enabled and the IPv6 neighbor discovery cache expiry is set to 7200 seconds (2 hours):

```
Device> enable
Device# configure terminal
Device(config)# interface Port-channel 189
Device(config-if)# no ip address
Device(config-if)# ipv6 address 2001:BD8::/64
Device(config-if)# ipv6 nd reachable-time 2700000
Device(config-if)# ipv6 nd na glean
Device(config-if)# ipv6 nd cache expire 7200
Device(config-if)# no ipv6 redirects
Device(config-if)# end
```



CHAPTER 60

IPv6 Neighbor Discovery Proxy

Prerequisites for IPv6 Neighbor Discovery Proxy

The following prerequisites are applicable when configuring IPv6 neighbor discovery proxy:

- Ensure that IPv6 is enabled on the Switch Virtual Interface (SVI).
- When you configure Duplicate Address Detection (DAD) proxy, ensure that device tracking is configured on the device.

Restrictions for IPv6 Neighbor Discovery Proxy

- IPv6 routing proxy is not supported on layer 3 interfaces.
- The IPv6 DAD proxy and routing proxy features are not supported on etherchannel ports.

Information About IPv6 Neighbor Discovery Proxy

IPv6 neighbor discovery proxy restricts IPv6 hosts within a VLAN from communicating directly with each other and allows them to communicate only via the gateway. A device operating as an IPv6 neighbor discovery proxy responds to packets on behalf of the target.

IPv6 neighbor discovery proxy operations are achieved using the following implementations:

IPv6 Routing-Proxy

A device operating as an IPv6 routing proxy listens to all neighbor discovery proxy messages sent on the link and responds unconditionally to neighbor solicitation lookup and neighbor-unreachability-detection messages

with neighbor advertisement (setting the SVI MAC address in the TLLA option) on behalf of the destination hosts to attract the traffic to itself.

IPv6 DAD Proxy

IPv6 DAD proxy feature responds to DAD queries on behalf of a node that owns the queried address. IPv6 DAD proxy depends on a device tracking database to ensure uniqueness of IPv6 addresses.

When receiving a DAD request from a host for a target, the DAD proxy performs a lookup into the binding table, and if the lookup returns a location, it sends an neighbor solicitation neighbor-unreachability-detection message to verify that the target is still alive.

- If the target replies to the neighbor-unreachability-detection message, the DAD proxy sends back an neighbor advertisement to the host (setting the SVI MAC address in the TLLA option).
- If the device does not respond to the neighbor-unreachability-detection message, the DAD proxy does not send any response to DAD request.

How to Configure IPv6 Neighbor Discovery Proxy

-

Configuring IPv6 Routing Proxy in VLAN Configuration Mode

Before you begin

Follow these steps to enable IPv6 on an SVI:

```
Device# enable
Device# configure terminal
Device(config)# interface vlan vlan-id
Device(config-if)# no ipv6 redirects
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 address ipv6-address
```

To configure IPv6 routing proxy in VLAN configuration mode, follow this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**vlan configuration** *vlan-id* **Example:** `Device(config)# vlan configuration 15`	Enters the VLAN configuration mode. This mode allows you to name, set the state, disable, and shut down the VLAN or range of VLANs.
Step 4	**[no] ipv6 nd routing-proxy** **Example:** `Device(config-vlan)# ipv6 nd routing-proxy`	Specifies if the neighbor discovery suppress must operate in routing proxy mode.
Step 5	**end** **Example:** `Device(config-vlan)# end`	Exits VLAN configuration mode and returns to privileged EXEC mode.

Configuring IPv6 Routing Proxy on an Interface

Before you begin

Follow these steps to enable IPv6 on an SVI:

```
Device# enable
Device# configure terminal
Device(config)# interface vlan vlan-id
Device(config-if)# no ipv6 redirects
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 address ipv6-address
```

To configure IPv6 routing proxy on an interface, follow this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet1/1`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**switchport access vlan** *vlan-id* **Example:** `Device(config)# switchport access vlan 15`	Assigns the port or range of ports into access ports.
Step 5	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Specifies which VLAN the interface belongs.
Step 6	**[no] ipv6 nd routing-proxy** **Example:** `Device(config-if)# ipv6 nd routing-proxy`	Specifies if the neighbor discovery suppress must operate in routing proxy mode.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring IPv6 DAD Proxy in VLAN Configuration Mode

Before you begin

- Follow these steps to enable IPv6 on an SVI:

```
Device# enable
Device# configure terminal
Device(config)# interface vlan vlan-id
Device(config-if)# no ipv6 redirects
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 address ipv6-address
```

- Attach a device tracking policy to the VLAN. For detailed steps, see the *Configuring Switch Integrated Security Features* chapter of the *Security Configuration Guide.*

To configure IPv6 DAD proxy in VLAN configuration mode, follow this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan configuration** *vlan-id* **Example:** `Device(config)# vlan configuration 15`	Enters the VLAN configuration mode. This mode allows you to name, set the state, disable, and shut down the VLAN or range of VLANs.
Step 4	**[no] ipv6 nd dad-proxy** **Example:** `Device(config-vlan)# ipv6 nd dad-proxy`	Specifies if the neighbor discovery suppress must operate in DAD proxy mode.
Step 5	**end** **Example:** `Device(config-vlan)# end`	Exits VLAN configuration mode and returns to privileged EXEC mode.

Configuring IPv6 DAD Proxy on an Interface

Before you begin

- Follow these steps to enable IPv6 on an SVI:

```
Device# enable
Device# configure terminal
Device(config)# interface vlan vlan-id
Device(config-if)# no ipv6 redirects
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 address ipv6-address
```

- Attach a device tracking policy to the layer 2 interface. For detailed steps, see the *Configuring Switch Integrated Security Features* chapter of the *Security Configuration Guide.*

To configure IPv6 DAD proxy on an interface, follow this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet1/11`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**switchport access vlan** *vlan-id* **Example:** `Device(config)# switchport access vlan 15`	Assigns the port or range of ports into access ports.
Step 5	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Specifies which VLAN the interface belongs.
Step 6	**[no] ipv6 nd dad-proxy** **Example:** `Device(config-if)# ipv6 nd dad-proxy`	Specifies if the neighbor discovery suppress must operate in DAD proxy mode.
Step 7	**end** **Example:**	Exits interface configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	

Verifying IPv6 Neighbor Discovery Proxy

Use the privileged EXEC or user EXEC commands in the table below to verify IPv6 neighbor discovery proxy information.

Table 61: Commands for Verifying IPv6 Neighbor Discovery Proxy

Commands	Description
show flooding-suppression	Displays flooding suppress policy (DAD proxy) configuration, and all the applied targets.
show ipv6 nd routing-proxy	Displays routing proxy default configuration, and all the applied targets .
show device-tracking policies	Displays device-tracking policy configuration, and all the applied targets.

Configuration Examples For IPv6 Neighbor Discovery Proxy

The following example shows the configuration of IPv6 routing proxy on a VLAN:

```
Device> enable
Device# configure terminal
Device(config)# vlan configuration  15
Device(config-vlan)# ipv6 nd routing-proxy
Device(config-vlan)# end
```

The following example shows the configuration of IPv6 DAD proxy on a VLAN:

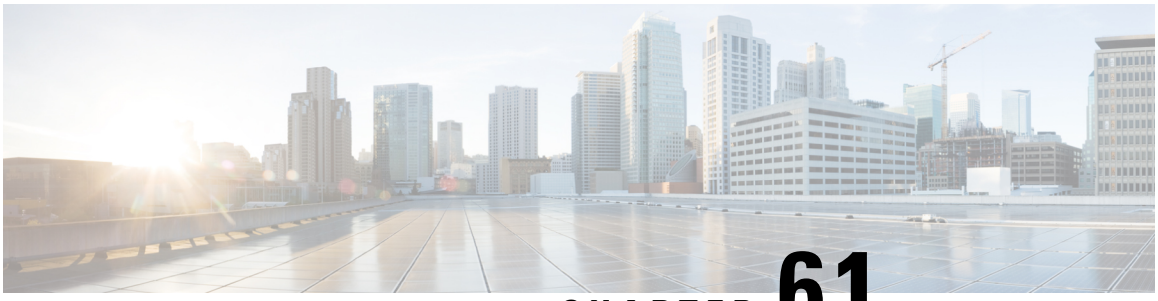
```
Device> enable
Device# configure terminal
Device(config)# vlan configuration  15
Device(config-vlan)# ipv6 nd dad-proxy
Device(config-vlan)# end
```

The following example shows the output of the **show flooding-suppression** command in privileged EXEC mode:

```
Device# show flooding-suppression

Flooding suppress policy DAD_PROXY configuration:
  Suppressing  NDP
mode:DAD proxy- RFC6957
Policy DAD_PROXY is applied on the following targets:
Target               Type  Policy              Feature        Target range
vlan 15             VLAN  DAD_PROXY           Flooding Suppress vlan all
```

The following example shows the output of the **show ipv6 nd routing-proxy**command in privileged EXEC mode:

```
Device# show ipv6 nd  routing-proxy

Routing Proxy default configuration:
  Proxying NDP
Policy default is applied on the following targets:
Target               Type  Policy              Feature        Target range
vlan 15             VLAN  default             Routing Proxy  vlan all
```



CHAPTER 61

IP Multicast Routing Technology Overview

Information About IP Multicast Technology

This section provides information about IP multicast technology.

About IP Multicast

At one end of the IP communication spectrum is IP unicast, where a source IP host sends packets to a specific destination IP host. In IP unicast, the destination address in the IP packet is the address of a single, unique host in the IP network. These IP packets are forwarded across the network from the source to the destination host by devices. At each point on the path between source and destination, a device uses a unicast routing table to make unicast forwarding decisions, based on the IP destination address in the packet.

At the other end of the IP communication spectrum is an IP broadcast, where a source host sends packets to all hosts on a network segment. The destination address of an IP broadcast packet has the host portion of the destination IP address set to all ones and the network portion set to the address of the subnet. IP hosts, including devices, understand that packets, which contain an IP broadcast address as the destination address, are addressed to all IP hosts on the subnet. Unless specifically configured otherwise, devices do not forward IP broadcast packets, so IP broadcast communication is normally limited to a local subnet.

IP multicasting falls between IP unicast and IP broadcast communication. IP multicast communication enables a host to send IP packets to a group of hosts anywhere within the IP network. To send information to a specific group, IP multicast communication uses a special form of IP destination address called an IP multicast group address. The IP multicast group address is specified in the IP destination address field of the packet.

To multicast IP information, Layer 3 switches and devices must forward an incoming IP packet to all output interfaces that lead to members of the IP multicast group.

We tend to think of IP multicasting and video conferencing as the same thing. Although the first application in a network to use IP multicast is often video conferencing, video is only one of many IP multicast applications that can add value to a company's business model. Other IP multicast applications that have potential for improving productivity include multimedia conferencing, data replication, real-time data multicasts, and simulation applications.

Role of IP Multicast in Information Delivery

IP multicast is a bandwidth-conserving technology that reduces traffic by delivering a single stream of information simultaneously to potentially thousands of businesses and homes. Applications that take advantage of multicast include video conferencing, corporate communications, distance learning, and distribution of software, stock quotes, and news.

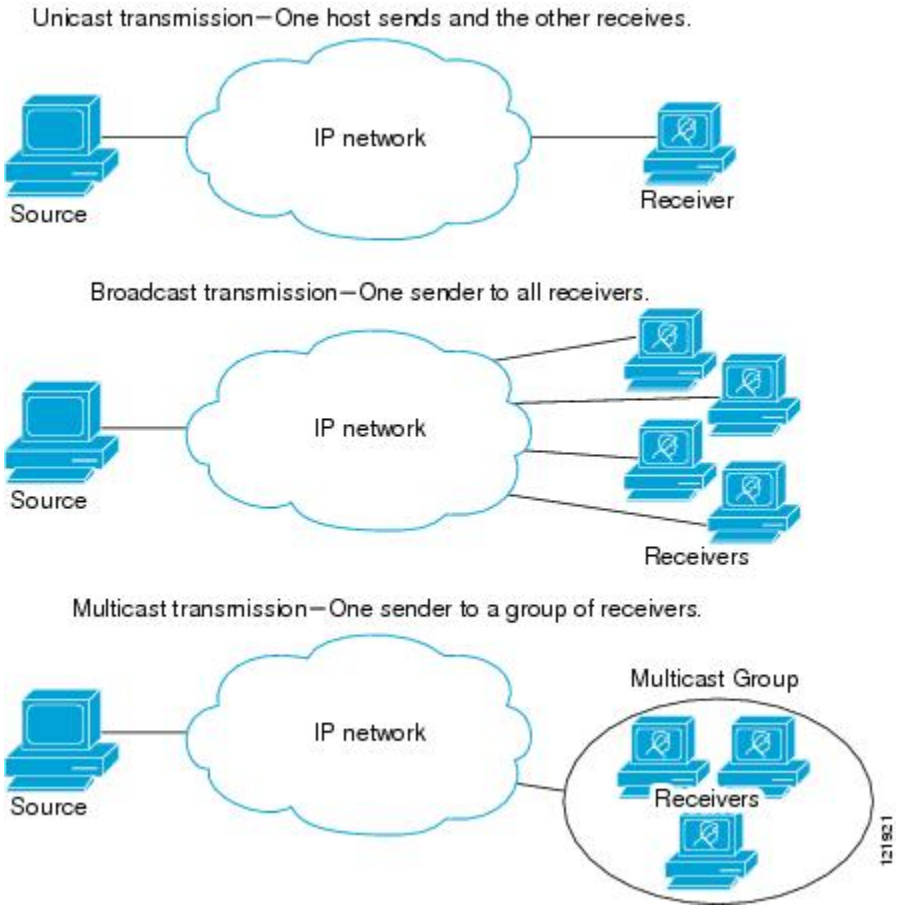
IP multicast routing enables a host (source) to send packets to a group of hosts (receivers) anywhere within the IP network by using a special form of IP address called the IP multicast group address. The sending host inserts the multicast group address into the IP destination address field of the packet and IP multicast routers and multilayer switches forward incoming IP multicast packets out all interfaces that lead to the members of the multicast group. Any host, regardless of whether it is a member of a group, can send to a group. However, only the members of a group receive the message. Controlling the transmission rate to a multicast group is not supported.

Multicast Group Transmission Scheme

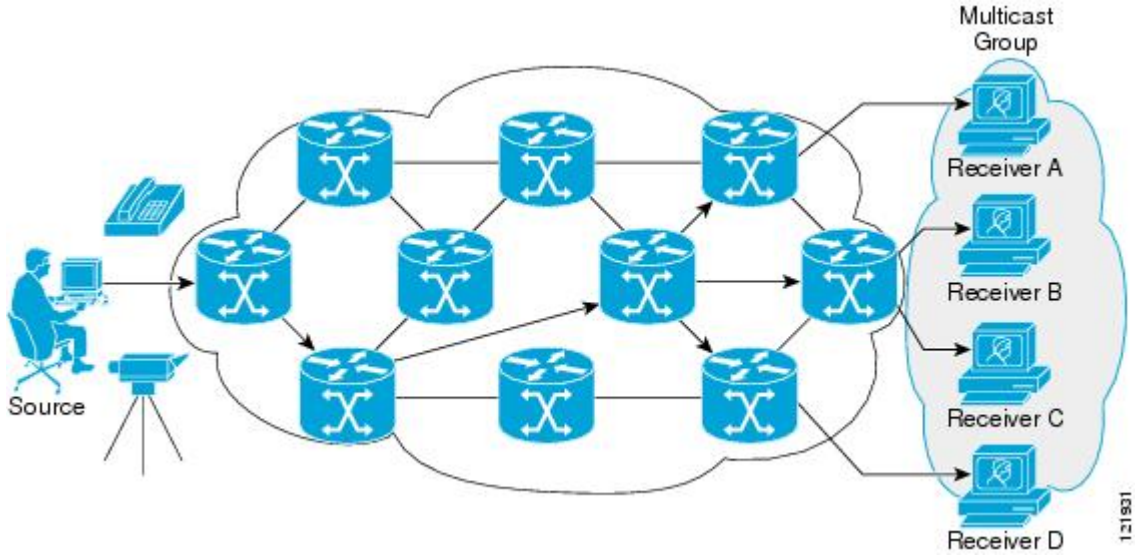
IP communication consists of hosts that act as senders and receivers of traffic as shown in the first figure. Senders are called sources. Traditional IP communication is accomplished by a single host source sending packets to another single host (unicast transmission) or to all hosts (broadcast transmission). IP multicast provides a third scheme, allowing a host to send packets to a subset of all hosts (multicast transmission). This subset of receiving hosts is called a multicast group. The hosts that belong to a multicast group are called group members.

Multicast is based on this group concept. A multicast group is an arbitrary number of receivers that join a group in order to receive a particular data stream. This multicast group has no physical or geographical boundaries--the hosts can be located anywhere on the Internet or on any private internetwork. Hosts that are interested in receiving data from a source to a particular group must join that group. Joining a group is accomplished by a host receiver by way of the Internet Group Management Protocol (IGMP).

In a multicast environment, any host, regardless of whether it is a member of a group, can send to a group. However, only the members of a group can receive packets sent to that group. Multicast packets are delivered to a group using best-effort reliability, just like IP unicast packets.



In the next figure, the receivers (the designated multicast group) are interested in receiving the video data stream from the source. The receivers indicate their interest by sending an IGMP host report to the routers in the network. The routers are then responsible for delivering the data from the source to the receivers. The routers use Protocol Independent Multicast (PIM) to dynamically create a multicast distribution tree. The video data stream will then be delivered only to the network segments that are in the path between the source and the receivers.



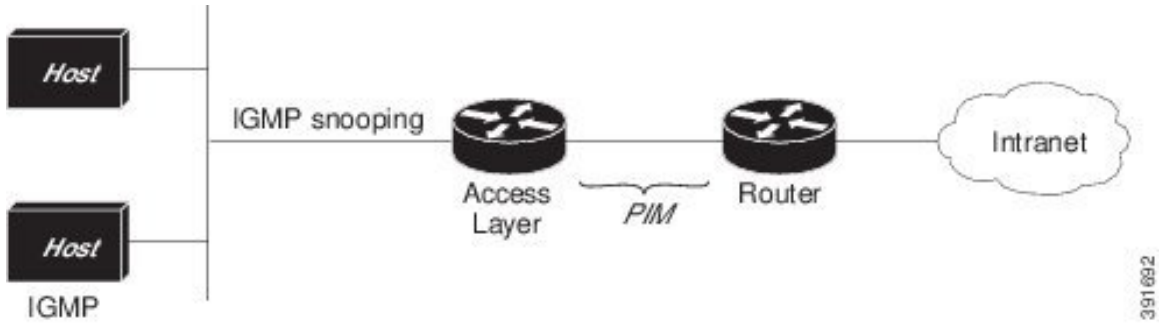
IP Multicast Routing Protocols

The software supports the following protocols to implement IP multicast routing:

- IGMP is used between hosts on a LAN and the routers (and multilayer devices) on that LAN to track the multicast groups of which hosts are members. To participate in IP multicasting, multicast hosts, routers, and multilayer devices must have the Internet Group Management Protocol (IGMP) operating.
- Protocol Independent Multicast (PIM) is used between routers so that they can track which multicast packets to forward to each other and to their directly connected LANs.
- IGMP Snooping is used for multicasting in a Layer 2 switching environment. It helps reduce the flooding of multicast traffic by dynamically configuring Layer 2 interfaces so that multicast traffic is forwarded to only those interfaces associated with IP multicast devices.

This figure shows where these protocols operate within the IP multicast environment.

Figure 63: IP Multicast Routing Protocols



According to IPv4 multicast standards, the MAC destination multicast address begins with 0100:5e and is appended by the last 23 bits of the IP address. For example, if the IP destination address is 239.1.1.39, the MAC destination address is 0100:5e01:0127.

A multicast packet is unmatched when the destination IPv4 address does not match the destination MAC address. The device forwards the unmatched packet in hardware based upon the MAC address table. If the destination MAC address is not in the MAC address table, the device floods the packet to the all port in the same VLAN as the receiving port.

Internet Group Management Protocol

IGMP messages are used by IP multicast hosts to send their local Layer 3 switch or router a request to join a specific multicast group and begin receiving multicast traffic. With some extensions in IGMPv2, IP hosts can also send a request to a Layer 3 switch or router to leave an IP multicast group and not receive the multicast group traffic.

Using the information obtained by using IGMP, a Layer 3 switch or router maintains a list of multicast group memberships on a per-interface basis. A multicast group membership is active on an interface if at least one host on the interface sends an IGMP request to receive multicast group traffic.

Protocol-Independent Multicast

Protocol-Independent Multicast (PIM) is protocol independent because it can leverage whichever unicast routing protocol is used to populate the unicast routing table, including EIGRP, OSPF, or static route, to support IP multicast.

PIM also uses a unicast routing table to perform the reverse path forwarding (RPF) check function instead of building a completely independent multicast routing table. PIM does not send and receive multicast routing updates between routers like other routing protocols do.

PIM Sparse Mode

PIM Sparse Mode (PIM-SM) uses a pull model to deliver multicast traffic. Only networks with active receivers that have explicitly requested the data are forwarded the traffic. PIM-SM is intended for networks with several different multicasts, such as desktop video conferencing and collaborative computing, that go to a small number of receivers and are typically in progress simultaneously.

Rendezvous Point

If you configure PIM to operate in sparse mode, you must also choose one or more devices to be rendezvous points (RPs). Senders to a multicast group use RPs to announce their presence. Receivers of multicast packets use RPs to learn about new senders. You can configure Cisco IOS software so that packets for a single multicast group can use one or more RPs.

The RP address is used by first hop devices to send PIM register messages on behalf of a host sending a packet to the group. The RP address is also used by last hop devices to send PIM join and prune messages to the RP to inform it about group membership. You must configure the RP address on all devices (including the RP device).

A PIM device can be an RP for more than one group. Only one RP address can be used at a time within a PIM domain for the same group. The conditions specified by the access list determine for which groups the device is an RP (as different groups can have different RPs).

IGMP Snooping

IGMP snooping is used for multicasting in a Layer 2 switching environment. With IGMP snooping, a Layer 3 switch or router examines Layer 3 information in the IGMP packets in transit between hosts and a device. When the switch receives the IGMP Host Report from a host for a particular multicast group, the switch adds

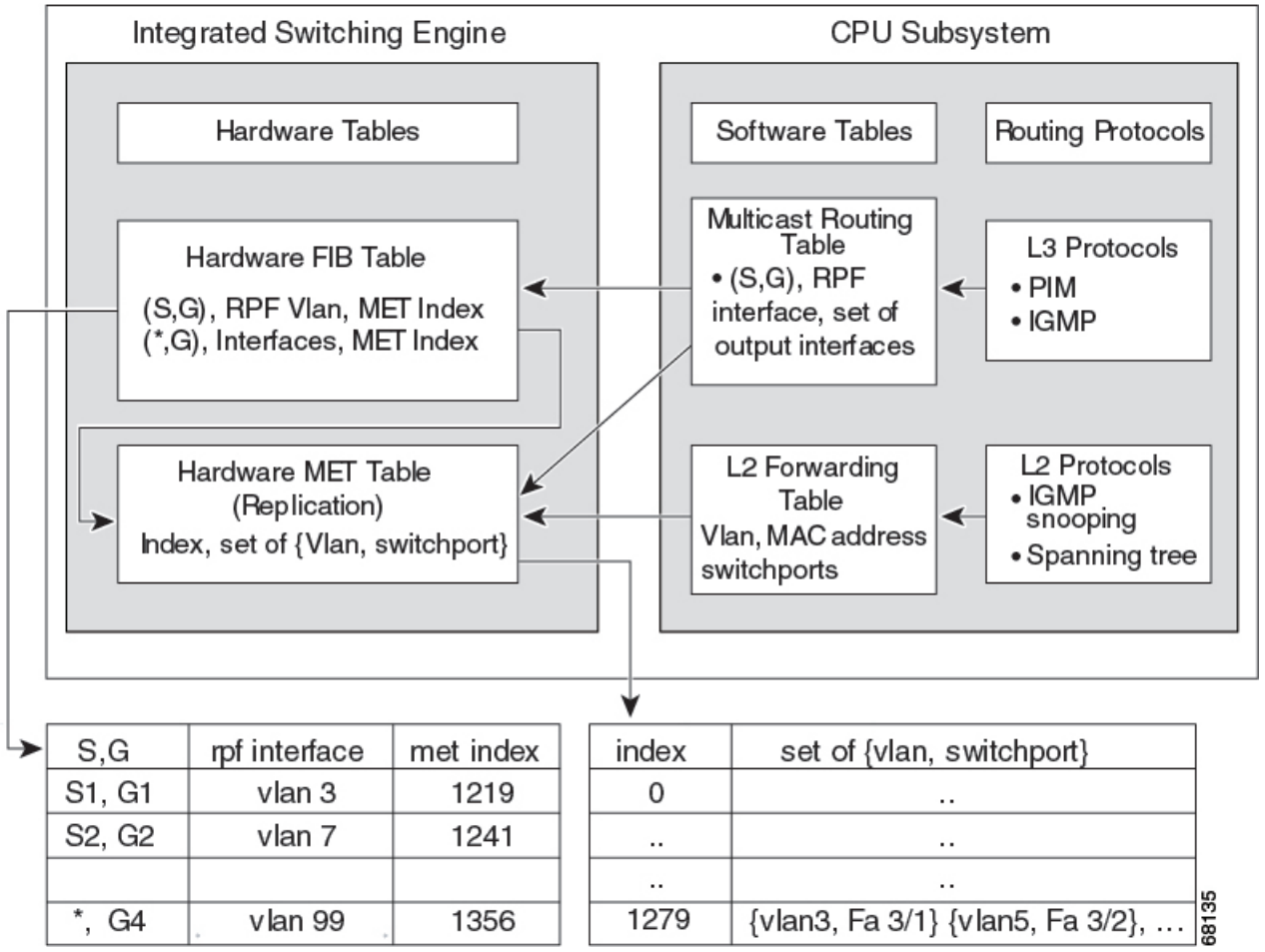
the host's port number to the associated multicast table entry. When the switch receives the IGMP Leave Group message from a host, it removes the host's port from the table entry.

Because IGMP control messages are transmitted as multicast packets, they are indistinguishable from multicast data if only the Layer 2 header is examined. A switch running IGMP snooping examines every multicast data packet to determine whether it contains any pertinent IGMP control information. If IGMP snooping is implemented on a low end switch with a slow CPU, performance could be severely impacted when data is transmitted at high rates.

IP Multicast Tables

The following illustration shows some key data structures that the device uses to forward IP multicast packets in hardware.

Figure 64: IP Multicast Tables and Protocols



The Integrated Switching Engine maintains the hardware FIB table to identify individual IP multicast routes. Each entry consists of a destination group IP address and an optional source IP address. Multicast traffic flows on primarily two types of routes: (S,G) and (*,G). The (S,G) routes flow from a source to a group based on the IP address of the multicast source and the IP address of the multicast group destination. Traffic on a (*,G) route flows from the PIM RP to all receivers of group G. Onlysparse-mode groups use (*,G) routes. The

Integrated Switching Engine hardware contains space for a total of 128,000 routes, which are shared by unicast routes, multicast routes, and multicast fast-drop entries.

Output interface lists are stored in the multicast expansion table (MET). The MET has room for up to 32,000 output interface lists. (For RET, we can have up to 102 K entries (32 K used for floodsets, 70,000 used for multicast entries)). The MET resources are shared by both Layer 3 multicast routes and by Layer 2 multicast entries. The actual number of output interface lists available in hardware depends on the specific configuration. If the total number of multicast routes exceed 32,000, multicast packets might not be switched by the Integrated Switching Engine. They would be forwarded by the CPU subsystem at much slower speeds.

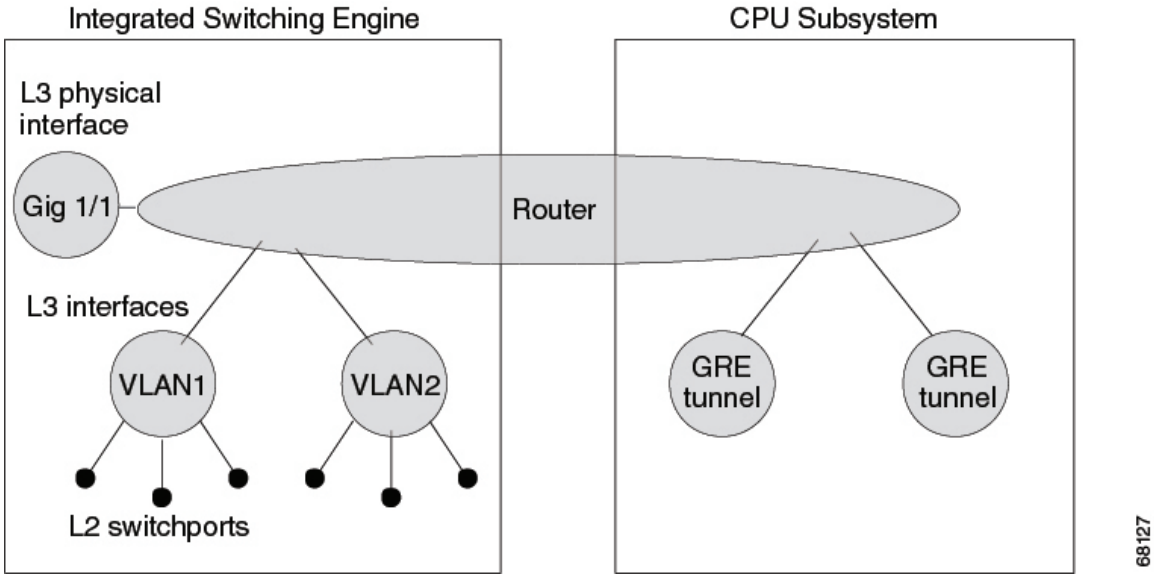
Note For RET, a maximum of 102 K entries is supported (32 K used for floodsets, 70 K used for multicast entries).

Hardware and Software Forwarding

The Integrated Switching Engine forwards the majority of packets in hardware at very high rates of speed. The CPU subsystem forwards exception packets in software. Statistical reports should show that the Integrated Switching Engine is forwarding the vast majority of packets in hardware.

The following illustration shows a logical view of the hardware and software forwarding components.

Figure 65: Hardware and Software Forwarding Components



In the normal mode of operation, the Integrated Switching Engine performs inter-VLAN routing in hardware. The CPU subsystem supports generic routing encapsulation (GRE) tunnels for forwarding in software.

Replication is a particular type of forwarding where, instead of sending out one copy of the packet, the packet is replicated and multiple copies of the packet are sent out. At Layer 3, replication occurs only for multicast packets; unicast packets are never replicated to multiple Layer 3 interfaces. In IP multicasting, for each incoming IP multicast packet that is received, many replicas of the packet are sent out.

IP multicast packets can be transmitted on the following types of routes:

- Hardware routes

- Software routes
- Partial routes

Hardware routes occur when the Integrated Switching Engine hardware forwards all replicas of a packet. Software routes occur when the CPU subsystem software forwards all replicas of a packet. Partial routes occur when the Integrated Switching Engine forwards some of the replicas in hardware and the CPU subsystem forwards some of the replicas in software.

Partial Routes

Note The conditions listed below cause the replicas to be forwarded by the CPU subsystem software, but the performance of the replicas that are forwarded in hardware is not affected.

The following conditions cause some replicas of a packet for a route to be forwarded by the CPU subsystem:

- The switch is configured with the **ip igmp join-group** command as a member of the IP multicast group on the RPF interface of the multicast source.
- The switch is the first-hop to the source in PIM sparse mode. The switch must send PIM-register messages to the RP.

Software Routes

Note If any one of the following conditions is configured on the RPF interface or the output interface, all replication of the output is performed in software.

The following conditions cause all replicas of a packet for a route to be forwarded by the CPU subsystem software:

- The interface is configured with multicast helper.
- The interface is a generic routing encapsulation (GRE) or Distance Vector Multicast Routing Protocol (DVMRP) tunnel.
- The interface uses non-Advanced Research Products Agency (ARPA) encapsulation.

The following packets are always forwarded in software:

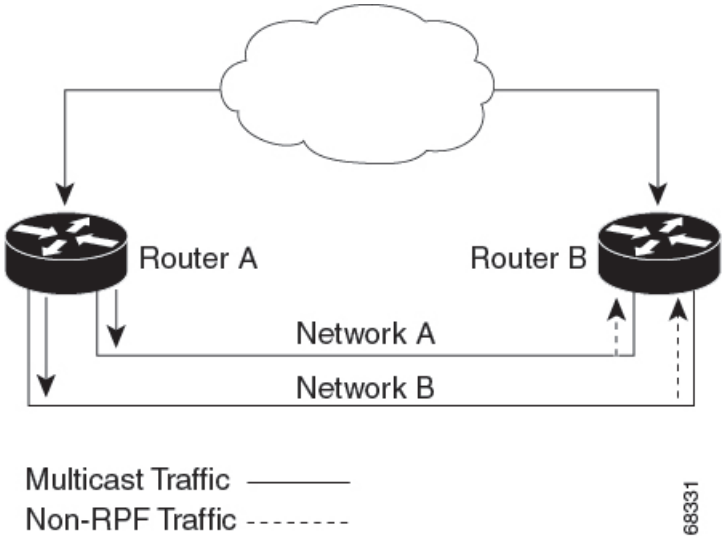
- Packets sent to multicast groups that fall into the range 224.0.0.* (where * is in the range from 0 to 255). This range is used by routing protocols. Layer 3 switching supports all other multicast group addresses.
- Packets with IP options.

Non-Reverse Path Forwarding Traffic

Traffic that fails an Reverse Path Forwarding (RPF) check is called non-RPF traffic. Non-RPF traffic is forwarded by the Integrated Switching Engine by filtering (persistently dropping) or rate limiting the non-RPF traffic.

In a redundant configuration where multiple Layer 3 switches or routers connect to the same LAN segment, only one device forwards the multicast traffic from the source to the receivers on the outgoing interfaces. The following illustration shows how non-RPF traffic can occur in a common network configuration.

Figure 66: Redundant Multicast Router Configuration in a Stub Network

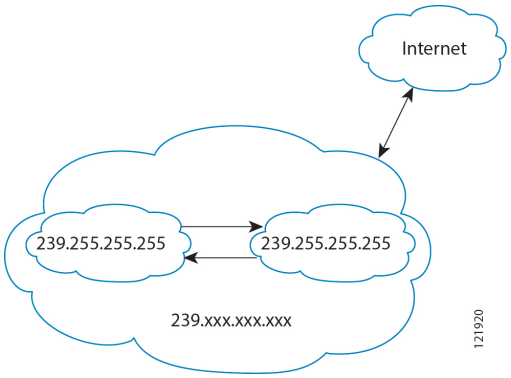


In this kind of topology, only Router A, the PIM designated router (PIM DR), forwards data to the common VLAN. Router B receives the forwarded multicast traffic, but must drop this traffic because it has arrived on the wrong interface and fails the RPF check. Traffic that fails the RPF check is called non-RPF traffic.

IP Multicast Boundary

As shown in the figure, address scoping defines domain boundaries so that domains with RPs that have the same IP address do not leak into each other. Scoping is performed on the subnet boundaries within large domains and on the boundaries between the domain and the Internet.

Figure 67: Address Scoping at Boundaries



You can set up an administratively scoped boundary on an interface for multicast group addresses using the **ip multicast boundary** command with the *access-list* argument. A standard access list defines the range of addresses affected. When a boundary is set up, no multicast data packets are allowed to flow across the

boundary from either direction. The boundary allows the same multicast group address to be reused in different administrative domains.

The Internet Assigned Numbers Authority (IANA) has designated the multicast address range 239.0.0.0 to 239.255.255.255 as the administratively scoped addresses. This range of addresses can be reused in domains administered by different organizations. They would be considered local, not globally unique.

You can configure the **filter-autorp** keyword to examine and filter Auto-RP discovery and announcement messages at the administratively scoped boundary. Any Auto-RP group range announcements from the Auto-RP packets that are denied by the boundary access control list (ACL) are removed. An Auto-RP group range announcement is permitted and passed by the boundary only if all addresses in the Auto-RP group range are permitted by the boundary ACL. If any address is not permitted, the entire group range is filtered and removed from the Auto-RP message before the Auto-RP message is forwarded.

IP Multicast Group Addressing

A multicast group is identified by its multicast group address. Multicast packets are delivered to that multicast group address. Unlike unicast addresses that uniquely identify a single host, multicast IP addresses do not identify a particular host. To receive the data sent to a multicast address, a host must join the group that address identifies. The data is sent to the multicast address and received by all the hosts that have joined the group indicating that they wish to receive traffic sent to that group. The multicast group address is assigned to a group at the source. Network administrators who assign multicast group addresses must make sure the addresses conform to the multicast address range assignments reserved by the Internet Assigned Numbers Authority (IANA).

IP Class D Addresses

IP multicast addresses have been assigned to the IPv4 Class D address space by IANA. The high-order four bits of a Class D address are 1110. Therefore, host group addresses can be in the range 224.0.0.0 to 239.255.255.255. A multicast address is chosen at the source (sender) for the receivers in a multicast group.

Note The Class D address range is used only for the group address or destination address of IP multicast traffic. The source address for multicast datagrams is always the unicast source address.

IP Multicast Address Scoping

The multicast address range is subdivided to provide predictable behavior for various address ranges and for address reuse within smaller domains. The table provides a summary of the multicast address ranges. A brief summary description of each range follows.

Table 62: Multicast Address Range Assignments

Name	Range	Description
Reserved Link-Local Addresses	224.0.0.0 to 224.0.0.255	Reserved for use by network protocols on a local network segment.
Globally Scoped Addresses	224.0.1.0 to 238.255.255.255	Reserved to send multicast data between organizations and across the Internet.

Name	Range	Description
Source Specific Multicast	232.0.0.0 to 232.255.255.255	Reserved for use with the SSM datagram delivery model where data is forwarded only to receivers that have explicitly joined the group.
GLOP Addresses	233.0.0.0 to 233.255.255.255	Reserved for statically defined addresses by organizations that already have an assigned autonomous system (AS) domain number.
Limited Scope Address	239.0.0.0 to 239.255.255.255	Reserved as administratively or limited scope addresses for use in private multicast domains.

Reserved Link-Local Addresses

The IANA has reserved the range 224.0.0.0 to 224.0.0.255 for use by network protocols on a local network segment. Packets with an address in this range are local in scope and are not forwarded by IP routers. Packets with link local destination addresses are typically sent with a time-to-live (TTL) value of 1 and are not forwarded by a router.

Within this range, reserved link-local addresses provide network protocol functions for which they are reserved. Network protocols use these addresses for automatic router discovery and to communicate important routing information. For example, Open Shortest Path First (OSPF) uses the IP addresses 224.0.0.5 and 224.0.0.6 to exchange link-state information.

IANA assigns single multicast address requests for network protocols or network applications out of the 224.0.1.xxx address range. Multicast routers forward these multicast addresses.

Note All the packets with reserved link-local addresses are punted to CPU by default in the ASR 903 RSP2 Module.

Globally Scoped Addresses

Addresses in the range 224.0.1.0 to 238.255.255.255 are called globally scoped addresses. These addresses are used to send multicast data between organizations across the Internet. Some of these addresses have been reserved by IANA for use by multicast applications. For example, the IP address 224.0.1.1 is reserved for Network Time Protocol (NTP).

Source Specific Multicast Addresses

Addresses in the range 232.0.0.0/8 are reserved for Source Specific Multicast (SSM) by IANA. In Cisco IOS software, you can use the **ip pim ssm**command to configure SSM for arbitrary IP multicast addresses also. SSM is an extension of Protocol Independent Multicast (PIM) that allows for an efficient data delivery mechanism in one-to-many communications. SSM is described in the IP Multicast Delivery Modes, on page 808 section.

GLOP Addresses

GLOP addressing (as proposed by RFC 2770, GLOP Addressing in 233/8) proposes that the 233.0.0.0/8 range be reserved for statically defined addresses by organizations that already have an AS number reserved. This practice is called GLOP addressing. The AS number of the domain is embedded into the second and third octets of the 233.0.0.0/8 address range. For example, AS 62010 is written in hexadecimal format as F23A.

Separating the two octets F2 and 3A results in 242 and 58 in decimal format. These values result in a subnet of 233.242.58.0/24 that would be globally reserved for AS 62010 to use.

Limited Scope Addresses

The range 239.0.0.0 to 239.255.255.255 is reserved as administratively or limited scoped addresses for use in private multicast domains. These addresses are constrained to a local group or organization. Companies, universities, and other organizations can use limited scope addresses to have local multicast applications that will not be forwarded outside their domain. Routers typically are configured with filters to prevent multicast traffic in this address range from flowing outside an autonomous system (AS) or any user-defined domain. Within an AS or domain, the limited scope address range can be further subdivided so that local multicast boundaries can be defined.

Note Network administrators may use multicast addresses in this range, inside a domain, without conflicting with others elsewhere in the Internet.

Layer 2 Multicast Addresses

Historically, network interface cards (NICs) on a LAN segment could receive only packets destined for their burned-in MAC address or the broadcast MAC address. In IP multicast, several hosts need to be able to receive a single data stream with a common destination MAC address. Some means had to be devised so that multiple hosts could receive the same packet and still be able to differentiate between several multicast groups. One method to accomplish this is to map IP multicast Class D addresses directly to a MAC address. Using this method, NICs can receive packets destined to many different MAC address.

Cisco Group Management Protocol (CGMP) is used on routers connected to switches to perform tasks similar to those performed by IGMP. CGMP is necessary for those switches that cannot distinguish between IP multicast data packets and IGMP report messages, both of which are addressed to the same group address at the MAC level.

Cisco Express Forwarding, MFIB, and Layer 2 Forwarding

The implementation of IP multicast is an extension of centralized Cisco Express Forwarding. Cisco Express Forwarding extracts information from the unicast routing table, which is created by unicast routing protocols, such as OSPF, and EIGRP and loads it into the hardware

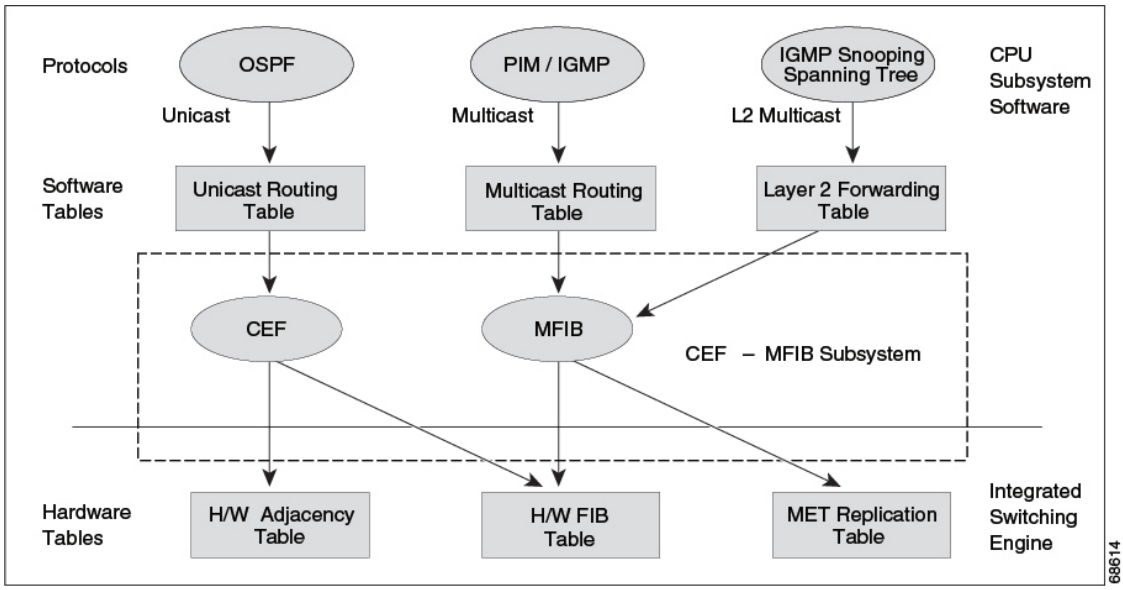
Forwarding Information Base (FIB). With the unicast routes in the FIB, when a route is changed in the upper-layer routing table, only one route needs to be changed in the hardware routing state. To forward unicast packets in hardware, the Integrated Switching Engine looks up source and destination routes in ternary content addressable memory (TCAM), takes the adjacency index from the hardware FIB, and gets the Layer 2 rewrite information and next-hop address from the hardware adjacency table.

The new Multicast Forwarding Information Base (MFIB) subsystem is the multicast analog of the unicast Cisco Express Forwarding. The MFIB subsystem extracts the multicast routes that PIM and IGMP create and refines them into a protocol-independent format for forwarding in hardware. The MFIB subsystem removes the protocol-specific information and leaves only the essential forwarding information. Each entry in the MFIB table consists of an (S,G) or (*,G) route, an input RPF VLAN, and a list of Layer 3 output interfaces. The MFIB subsystem, together with platform-dependent management software, loads this multicast routing

information into the hardware FIB and Replica Expansion Table (RET). The device performs Layer 3 routing and Layer 2 bridging at the same time. There can be multiple Layer 2 switch ports on any VLAN interface.

The following illustration shows a functional overview of how a Cisco device combines unicast routing, multicast routing, and Layer 2 bridging information to forward in hardware:

Figure 68: Combining Cisco Express Forwarding, MFIB, and Layer 2 Forwarding Information in Hardware



Like the Cisco Express Forwarding unicast routes, the MFIB routes are Layer 3 and must be merged with the appropriate Layer 2 information. The following example shows an MFIB route:

```
(*,203.0.113.1)
RPF interface is Vlan3
Output Interfaces are:
Vlan 1
Vlan 2
```

The route (*,203.0.113.1) is loaded in the hardware FIB table and the list of output interfaces is loaded into the MET. A pointer to the list of output interfaces, the MET index, and the RPF interface are also loaded in the hardware FIB with the (*,203.0.113.1) route. With this information loaded in hardware, merging of the Layer 2 information can begin. For the output interfaces on VLAN1, the Integrated Switching Engine must send the packet to all switch ports in VLAN1 that are in the spanning tree forwarding state. The same process applies to VLAN 2. To determine the set of switch ports in VLAN 2, the Layer 2 Forwarding Table is used.

When the hardware routes a packet, in addition to sending it to all of the switch ports on all output interfaces, the hardware also sends the packet to all switch ports (other than the one it arrived on) in the input VLAN. For example, assume that VLAN 3 has two switch ports in it, GigabitEthernet 3/1 and GigabitEthernet 3/2. If a host on GigabitEthernet 3/1 sends a multicast packet, the host on GigabitEthernet 3/2 might also need to receive the packet. To send a multicast packet to the host on GigabitEthernet 3/2, all of the switch ports in the ingress VLAN must be added to the port set that is loaded in the MET.

If VLAN 1 contains 1/1 and 1/2, VLAN 2 contains 2/1 and 2/2, and VLAN 3 contains 3/1 and 3/2, the MET chain for this route would contain these switch ports: (1/1,1/2,2/1,2/2,3/1, and 3/2).

If IGMP snooping is on, the packet should not be forwarded to all output switch ports on VLAN 2. The packet should be forwarded only to switch ports where IGMP snooping has determined that there is either a group

member or router. For example, if VLAN 1 had IGMP snooping enabled, and IGMP snooping determined that only port 1/2 had a group member on it, then the MET chain would contain these switch ports: (1/1,1/2, 2/1, 2/2, 3/1, and 3/2).

IP Multicast Delivery Modes

IP multicast delivery modes differ only for the receiver hosts, not for the source hosts. A source host sends IP multicast packets with its own IP address as the IP source address of the packet and a group address as the IP destination address of the packet.

Source Specific Multicast

Source Specific Multicast (SSM) is a datagram delivery model that best supports one-to-many applications, also known as broadcast applications. SSM is a core network technology for the Cisco implementation of IP multicast targeted for audio and video broadcast application environments.

For the SSM delivery mode, an IP multicast receiver host must use IGMP Version 3 (IGMPv3) to subscribe to channel (S,G). By subscribing to this channel, the receiver host is indicating that it wants to receive IP multicast traffic sent by source host S to group G. The network will deliver IP multicast packets from source host S to group G to all hosts in the network that have subscribed to the channel (S, G).

SSM does not require group address allocation within the network, only within each source host. Different applications running on the same source host must use different SSM groups. Different applications running on different source hosts can arbitrarily reuse SSM group addresses without causing any excess traffic on the network.

Multicast Fast Drop

In IP multicast protocols, such as PIM-SM and PIM-DM, every (S,G) or (*,G) route has an incoming interface associated with it. This interface is referred to as the reverse path forwarding interface. In some cases, when a packet arrives on an interface other than the expected RPF interface, the packet must be forwarded to the CPU subsystem software to allow PIM to perform special protocol processing on the packet. One example of this special protocol processing that PIM performs is the PIM Assert protocol.

By default, the Integrated Switching Engine hardware sends all packets that arrive on a non-RPF interface to the CPU subsystem software. However, processing in software is not necessary in many cases, because these non-RPF packets are often not needed by the multicast routing protocols. The problem is that if no action is taken, the non-RPF packets that are sent to the software can overwhelm the CPU.

Instead of installing fast-drop entries, the Cisco device uses Dynamic Buffer Limiting (DBL). This flow-based congestion avoidance mechanism provides active queue management by tracking the queue length for each traffic flow. When the queue length of a flow exceeds its set limit, DBL drops packets. Rate DBL limits the non-rpf traffic to the cpu subsystem so that the CPU is not overwhelmed. The packets are rate limited per flow to the CPU. Because installing fast-drop entries in the CAM is inaccessibly, the number of fast-drop flows that can be handled by the switch need not be limited.

Protocol events, such as a link going down or a change in the unicast routing table, can impact the set of packets that can safely be fast dropped. A packet that was correctly fast dropped before might, after a topology change, need to be forwarded to the CPU subsystem software so that PIM can process it. The CPU subsystem software handles flushing fast-drop entries in response to protocol events so that the PIM code in IOS can process all the necessary RPF failures.

The use of fast-drop entries in the hardware is critical in some common topologies because you may have persistent RPF failures. Without the fast-drop entries, the CPU is exhausted by RPF failed packets that it did not need to process.

Multicast Forwarding Information Base

The Multicast Forwarding Information Base (MFIB) subsystem supports IP multicast routing in the Integrated Switching Engine hardware on Cisco devices. The MFIB logically resides between the IP multicast routing protocols in the CPU subsystem software (PIM, IGMP, MSDP, and DVMRP) and the platform-specific code that manages IP multicast routing in hardware. The MFIB translates the routing table information created by the multicast routing protocols into a simplified format that can be efficiently processed and used for forwarding by the Integrated Switching Engine hardware.

To display the information in the multicast routing table, use the **show ip mroute** command. To display the MFIB table information, use the **show ip mfib** command.

The MFIB table contains a set of IP multicast routes. IP multicast routes include (S,G) and (*,G). Each route in the MFIB table can have one or more optional flags associated with it. The route flags indicate how a packet that matches a route should be forwarded. For example, the Internal Copy (IC) flag on an MFIB route indicates that a process on the switch needs to receive a copy of the packet. The following flags can be associated with MFIB routes:

- Internal Copy (IC) flag—Sets on a route when a process on the router needs to receive a copy of all packets matching the specified route.
- Signalling (S) flag—Sets on a route when a process needs to be notified when a packet matching the route is received; the expected behavior is that the protocol code updates the MFIB state in response to receiving a packet on a signalling interface.
- Connected (C) flag—When set on an MFIB route, has the same meaning as the Signaling (S) flag, except that the C flag indicates that only packets sent by directly connected hosts to the route should be signaled to a protocol process.

A route can also have a set of optional flags associated with one or more interfaces. For example, an (S,G) route with the flags on VLAN 1 indicates how packets arriving on VLAN 1 should be handled, and whether packets matching the route should be forwarded onto VLAN 1. The per-interface flags supported in the MFIB include the following:

- Accepting (A)—Sets on the interface that is known in multicast routing as the RPF interface. A packet that arrives on an interface that is marked as Accepting (A) is forwarded to all Forwarding (F) interfaces.
- Forwarding (F)—Used in conjunction with the Accepting (A) flag as described above. The set of Forwarding interfaces that form what is often referred to as the multicast “olist” or output interface list.
- Signaling (S)—Sets on an interface when some multicast routing protocol process in Cisco IOS needs to be notified of packets arriving on that interface.

Note When PIM-SM routing is in use, the MFIB route might include an interface as in this example:

```
PimTunnel [1.2.3.4]
```

It is a virtual interface that the MFIB subsystem creates to indicate that packets are being tunnelled to the specified destination address. A PimTunnel interface cannot be displayed with the normal **show interface** command.

Subnet/Mask Length

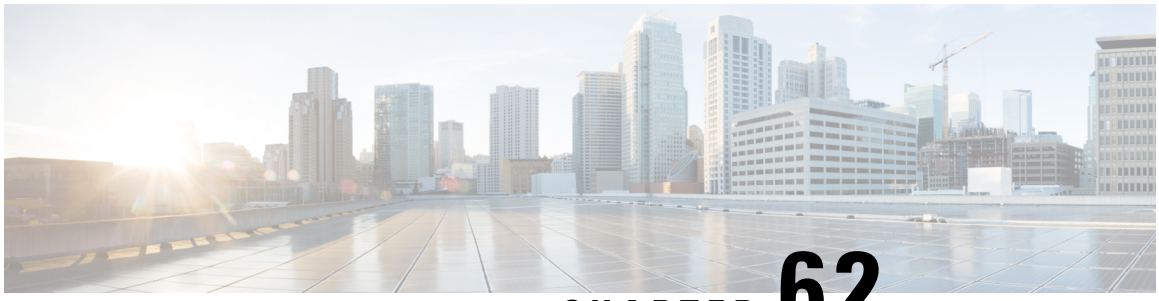
An (S/M, 224/4) entry is created in the MFIB for every multicast-enabled interface. This entry ensures that all packets sent by directly connected neighbors can be register-encapsulated to the PIM-SM RP. Typically, only a small number of packets are forwarded using the (S/M,224/4) route, until the (S,G) route is established by PIM-SM.

For example, on an interface with IP address 10.0.0.1 and netmask 255.0.0.0, a route is created matching all IP multicast packets in which the source address is anything in the class A network 10. This route can be written in conventional subnet/masklength notation as (10/8,224/4). If an interface has multiple assigned IP addresses, then one route is created for each such IP address.

Multicast High Availability

The switch supports multicast high availability, which ensures uninterrupted multicast traffic flow if a supervisor engine failure. MFIB states are synced to the standby supervisor engine before a switchover.

Multicast HA (ISSU) is supported for the PIM Sparse mode and SSM mode; and in Layer 2 for IGMP and MLD Snooping.



CHAPTER 62

Configuring Basic IP Multicast Routing

Information About Basic IP Multicast Routing

IP multicasting is an efficient way to use network resources, especially for bandwidth-intensive services such as audio and video. IP multicast routing enables a host (source) to send packets to a group of hosts (receivers) anywhere within the IP network by using a special form of IP address called the IP multicast group address.

The sending host inserts the multicast group address into the IP destination address field of the packet, and IP multicast routers and multilayer switches forward incoming IP multicast packets out all interfaces that lead to members of the multicast group. Any host, regardless of whether it is a member of a group, can send to a group. However, only the members of a group receive the message.

Multicast Forwarding Information Base Overview

The device uses the Multicast Forwarding Information Base (MFIB) architecture and the Multicast Routing Information Base (MRIB) for IP multicast.

The MFIB architecture provides both modularity and separation between the multicast control plane (Protocol Independent Multicast [PIM] and Internet Group Management Protocol [IGMP]) and the multicast forwarding plane (MFIB). This architecture is used in Cisco IOS IPv6 multicast implementations.

MFIB itself is a multicast routing protocol independent forwarding engine; that is, it does not depend on PIM or any other multicast routing protocol. It is responsible for:

- Forwarding multicast packets
- Registering with the MRIB to learn the entry and interface flags set by the control plane
- Handling data-driven events that must be sent to the control plane
- Maintaining counts, rates, and bytes of received, dropped, and forwarded multicast packets

The MRIB is the communication channel between MRIB clients. Examples of MRIB clients are PIM, IGMP, the multicast routing (mroute) table, and the MFIB.

Default IP Multicast Routing Configuration

This table displays the default IP multicast routing configuration.

Table 63: Default IP Multicast Routing Configuration

Feature	Default Setting
Multicast routing	Disabled on all interfaces.
PIM version	Version 2.
PIM mode	No mode is defined.
PIM stub routing	None configured.
PIM RP address	None configured.
PIM domain border	Disabled.
PIM multicast boundary	None.
Candidate BSRs	Disabled.
Candidate RPs	Disabled.
Shortest-path tree threshold rate	0 kb/s.
PIM router query message interval	30 seconds.

How to Configure Basic IP Multicast Routing

This section provides information about configuring basic IP multicast routing.

Configuring Basic IP Multicast Routing

Before you begin

Note By default, multicast routing is disabled, and there is no default mode setting. To enable multicast routing, use the **ip multicast-routing** command.

You must configure the PIM version and the PIM mode. The switch populates its multicast routing table and forwards multicast packets it receives from its directly connected LANs according to the mode setting.

In populating the multicast routing table, dense-mode interfaces are always added to the table. Sparse-mode interfaces are added to the table only when periodic join messages are received from downstream devices or when there is a directly connected member on the interface. When forwarding from a LAN, sparse-mode operation occurs if there is an RP known for the group. If so, the packets are encapsulated and sent toward the RP. When no RP is known, the packet is flooded in a dense-mode fashion. The multicast source address

must be on the directly connected incoming interface (that is part of the same subnet) of the first-hop router for both PIM dense mode and PIM any-source multicast mode. If the multicast traffic from a specific source is sufficient, the receiver's first-hop router might send join messages toward the source to build a source-based distribution tree.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the Layer 3 interface on which you want to enable multicast routing, and enters interface configuration mode. The specified interface must be one of the following: • A routed port—A physical port that has been configured as a Layer 3 port by entering the **no switchport** interface configuration command. You will also need to enable IP PIM sparse-dense-mode on the interface, and join the interface as a statically connected member to an IGMP static group. • An SVI—A VLAN interface created by using the **interface vlan** *vlan-id* global configuration command. You will also need to enable IP PIM sparse-dense-mode on the VLAN, join the VLAN as a statically connected member to an IGMP static group, and then enable IGMP snooping on the VLAN, the IGMP static group, and physical interface. These interfaces must have IP addresses assigned to them.
Step 4	**ip pim** {**dense-mode** | **sparse-mode** | **sparse-dense-mode**}	Enables a PIM mode on the interface. By default, no mode is configured.

	Command or Action	Purpose
	Example: `Device(config-if)# ip pim sparse-mode`	The keywords have these meanings: • **dense-mode**—Enables dense mode of operation. • **sparse-mode**—Enables sparse mode of operation. If you configure sparse mode, you must also configure an RP. • **sparse-dense-mode**—Causes the interface to be treated in the mode in which the group belongs. Sparse-dense mode is the recommended setting. **Note** To disable PIM on an interface, use the **no ip pim** interface configuration command.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring IP Multicast Forwarding

You can use the following procedure to configure IPv4 Multicast Forwarding Information Base (MFIB) interrupt-level IP multicast forwarding of incoming packets or outgoing packets on the device.

Note After you have enabled IP multicast routing by using the **ip multicast-routing** command, IPv4 multicast forwarding is enabled.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip mfib** **Example:** `Device(config)# ip mfib`	Enables IP multicast forwarding.
Step 4	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Static Multicast Route (mroute)

- Static mroutes are used to calculate RPF information, not to forward traffic.
- Static mroutes cannot be redistributed.

Static mroutes are strictly local to the device on which they are defined. Because Protocol Independent Multicast (PIM) does not have its own routing protocol, there is no mechanism to distribute static mroutes throughout the network. Consequently, the administration of static mroutes tends to be more complicated than the administration of unicast static routes.

When static mroutes are configured, they are stored on the device in a separate table referred to as the static mroute table. When configured, the **ip mroute** command enters a static mroute into the static mroute table for the source address or source address range specified for the source-address and mask arguments. Sources that match the source address or that fall in the source address range specified for the source-address argument will RPF to either the interface associated with the IP address specified for the *rpf-address* argument or the local interface on the device specified for the *interface-type* and *interface-number* arguments. If an IP address is specified for the *rpf-address* argument, a recursive lookup is done from the unicast routing table on this address to find the directly connected neighbor.

If there are multiple static mroutes configured, the device performs a longest-match lookup of the mroute table. When the mroute with the longest match (of the source-address) is found, the search terminates and the information in the matching static mroute is used. The order in which the static mroutes are configured is not important.

The administrative distance of an mroute may be specified for the optional distance argument. If a value is not specified for the distance argument, the distance of the mroute defaults to zero. If the static mroute has the same distance as another RPF source, the static mroute will take precedence. There are only two exceptions to this rule: directly connected routes and the default unicast route.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip mroute** [**vrf** *vrf-name*] *source-address mask* { **fallback-lookup** {**global** | **vrf** *vrf-name* }[*protocol*] {*rpf-address* | *interface-type interface-number*}} [**distance**] **Example:** `Device(config)# ip mroute 10.1.1.1 255.255.255.255 10.2.2.2`	The source IP address 10.1.1.1 is configured to be reachable through the interface associated with IP address 10.2.2.2.
Step 4	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	(Optional) Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Multicast VRFs

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 2	**ip routing** **Example:** `Switch(config)# ip routing`	Enables IP routing.
Step 3	**ip vrf** *vrf-name* **Example:** `Switch(config)# ip vrf vrf-name`	Configures a VRF table and enters VRF configuration mode.
Step 4	**ip multicast-routing vrf** *vrf-name* **Example:** `Switch(config-vrf)# ip multicast-routing vrf vrf-name`	(Optional) Enables global multicast routing for VRF table.
Step 5	**rd** *route-distinguisher* **Example:** `Switch (config-vrf)# rd route-distinguisher`	Creates a VRF table by specifying a route distinguisher. Enter either an AS number and an arbitrary number (xxx:y) or an IP address and arbitrary number (A.B.C.D:y).
Step 6	**route-target** {**export** | **import** | **both**} *route-target-ext-community* **Example:** `Switch(config-vrf)# route-target {export | import | both} route-target-ext-community`	Creates a list of import, export, or import and export route target communities for the specified VRF. Enter either an AS system number and an arbitrary number (xxx:y) or an IP address and an arbitrary number (A.B.C.D:y). The route-target-ext-community value should be the same as the route-distinguisher value entered in Step 4.
Step 7	**import map** *route-map* **Example:** `Switch(config-vrf)# import map route-map`	(Optional) Associates a route map with the VRF.
Step 8	**interface** *interface-id* **Example:** `Switch (config)# interface interface-id`	Enters interface configuration mode and specifies the Layer 3 interface to be associated with the VRF. The interface can be a routed port or a SVI.

	Command or Action	Purpose
Step 9	**vrf forwarding** *vrf-name* **Example:** `Switch (config-sg-tacacs+)# vrf forwarding vrf-name`	Associates the VRF with the Layer 3 interface.
Step 10	**ip address** *ip-address* **mask** **Example:** `Switch (config-if)# ip address ip-address mask`	Configures IP address for the Layer 3 interface.
Step 11	**ip pim sparse-mode** **Example:** `Switch(config-if)# ip pim sparse-mode`	Enables PIM on the VRF-associated Layer 3 interface.
Step 12	**end** **Example:** `Switch(config-if)# end`	Returns to privileged EXEC mode.
Step 13	show ip vrf [brief | detail | interfaces] [vrf-name] **Example:** **show ip vrf** [**brief** | **detail** | **interfaces**] [*vrf-name*]	Verifies the configuration. Display information about the configured VRFs.
Step 14	**copy running-config startup-config** **Example:** `Switch# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Example

The following example shows how to configure multicast within a VRF table:

```
Switch(config)# ip routing
Switch(config)# ip vrf multiVrfA
Switch(config-vrf)# ip multicast-routing vrf multiVrfA
Switch(config-vrf)# interface GigabitEthernet3/1
Switch(config-if)# vrf forwarding multiVrfA
Switch(config-if)# ip address 172.21.200.203 255.255.255.0
Switch(config-if)# ip pim sparse-mode
```

Configuring Optional IP Multicast Routing Features

This section provides information about configuringg optional IP multicast routing features.

Defining the IP Multicast Boundary

You define a multicast boundary to prevent Auto-RP messages from entering the PIM domain. You create an access list to deny packets destined for 224.0.1.39 and 224.0.1.40, which carry Auto-RP information.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *access-list-number* **deny** *source* [*source-wildcard*] **Example:** `Device(config)#` `access-list 12 deny 224.0.1.39` `access-list 12 deny 224.0.1.40`	Creates a standard access list, repeating the command as many times as necessary. • For *access-list-number*, the range is 1 to 99. • The **deny** keyword denies access if the conditions are matched. • For *source*, enter multicast addresses 224.0.1.39 and 224.0.1.40, which carry Auto-RP information. • (Optional) For *source-wildcard*, enter the wildcard bits in dotted decimal notation to be applied to the source. Place ones in the bit positions that you want to ignore. The access list is always terminated by an implicit deny statement for everything.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be configured, and enters interface configuration mode.
Step 5	**ip multicast boundary** *access-list-number* **Example:** `Device(config-if)# ip multicast boundary 12`	Configures the boundary, specifying the access list you created in Step 2.
Step 6	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring sdr Listener Support

This section provides information about configuring sdr listener support.

Enabling sdr Listener Support

By default, the device does not listen to session directory advertisements. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be enabled for sdr, and enters interface configuration mode.
Step 4	**ip sap listen** **Example:** `Device(config-if)# ip sap listen`	Enables the device software to listen to session directory announcements.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:**	Verifies your entries.

	Command or Action	Purpose
	`Device# show running-config`	
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Limiting How Long an sdr Cache Entry Exists

By default, entries are never deleted from the sdr cache. You can limit how long the entry remains active so that if a source stops advertising SAP information, old advertisements are not unnecessarily kept.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip sap cache-timeout** *minutes* **Example:** `Device(config)# ip sap cache-timeout 30`	Limits how long a Session Announcement Protocol (SAP) cache entry stays active in the cache. By default, entries are never deleted from the cache. For *minutes*, the range is 1 to 1440 minutes (24 hours).
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**show ip sap** **Example:** `Device# show ip sap`	Displays the SAP cache.

	Command or Action	Purpose
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring and Maintaining Basic IP Multicast Routing

Clearing Caches, Tables, and Databases

You can remove all contents of a particular cache, table, or database. Clearing a cache, table, or database might be necessary when the contents of the particular structure are or suspected to be invalid.

You can use any of the privileged EXEC commands in the following table to clear IP multicast caches, tables, and databases.

Table 64: Commands for Clearing Caches, Tables, and Databases

Command	Purpose
clear ip igmp group {**group** [*hostname* | *IP address*] | **vrf** *name* **group** [*hostname* | *IP address*] }	Deletes entries from the IGMP cache.
clear ip mroute { ***** | [*hostname* | *IP address*] | **vrf** *name* **group** [*hostname* | *IP address*] }	Deletes entries from the IP multicast routing table.
clear ip sap [*group-address* | **"***session-name***"**]	Deletes the Session Directory Protocol Version 2 cache or an sdr cache entry.

Displaying System and Network Statistics

You can display specific statistics, such as the contents of IP routing tables, caches, and databases.

Note This release does not support per-route statistics.

You can display information to learn resource usage and solve network problems. You can also display information about node reachability and discover the routing path that packets of your device are taking through the network.

You can use any of the privileged EXEC commands in the following table to display various routing statistics.

Table 65: Commands for Displaying System and Network Statistics

Command	Purpose
ping [*group-name* | *group-address*]	Sends an ICMP Echo Request to a multicast group address.
show ip igmp filter	Displays IGMP filter information.
show ip igmp groups [*group-name*|*group-address*|*type-number*]	Displays the multicast groups that are directly connected to the device and that were learned through IGMP.
show ip igmp interface [*type number*]	Displays multicast-related information about an interface.
show ip igmp profile [profile_number]	Displays IGMP profile information.
show ip igmp ssm-mapping [*hostname/IP address*]	Displays IGMP SSM mapping information.
show ip igmp static-group {**class-map** [**interface** [type]]	Displays static group information.
show ip igmp membership [*name/group address* | **all** | **tracked**]	Displays IGMP membership information for forwarding.
show ip igmp vrf	Displays the selected VPN Routing/Forwarding instance by name.
show ip mfib [*type number*]	Displays the IP multicast forwarding information base.
show ip mrib { **client** | **route** | **vrf** }	Displays the multicast routing information base.
show ip mrm { **interface** | **manager** | **status-report** }	Displays the IP multicast routing monitor information.
show ip mroute [*group-name* | *group-address*] [*source*] [**count** | **interface** | **proxy** | **pruned** | **summary** | **verbose**]	Displays the contents of the IP multicast routing table.
show ip msdp { **count** | **peer** | **rpf-peer** | **sa-cache** | **summary** | **vrf** }	Displays the Multicast Source Discovery Protocol (MSDP) information.
show ip multicast [**interface** | **limit** | **mpls** | **redundancy** | **vrf**]	Displays global multicast information.
show ip pim all-vrfs { **tunnel** }	Display all VRFs.
show ip pim autorp	Display global auto-RP information.
show ip pim boundary [*type number*]	Displays boundary information.

Command	Purpose
show ip pim bsr-router	Display bootstrap router information (version 2).
show ip pim interface [*type number*] [**count** | **detail** | **df** | **stats**]	Displays information about interfaces configured for PIM. This command is available in all software images.
show ip pim neighbor [*type number*]	Lists the PIM neighbors discovered by the device. This command is available in all software images.
show ip pim mdt [**bgp**]	Displays multicast tunnel information.
show ip pim rp [*group-name* | *group-address*]	Displays the RP routers associated with a sparse-mode multicast group. This command is available in all software images.
show ip pim rp-hash [*group-name* | *group-address*]	Displays the RP to be chosen based upon the group selected.
show ip pim tunnel [*tunnel* | *verbose*]	Displays the registered tunnels.
show ip pim vrf *name*	Displays VPN routing and forwarding instances.
show ip rpf {*source-address* | *name*}	Displays how the device is doing Reverse-Path Forwarding (that is, from the unicast routing table, DVMRP routing table, or static mroutes). Command parameters include: • *Host name* or *IP address*—IP name or group address. • **Select**—Group-based VRF select information. • **vrf**—Selects VPN Routing/Forwarding instance.
show ip sap [*group* | **"***session-name***"** | **detail**]	Displays the Session Announcement Protocol (SAP) Version 2 cache. Command parameters include: • *A.B.C.D*—IP group address. • *WORD*—Session name (in double quotes). • **detail**—Session details.

Configuration Examples for Basic IP Multicast Routing

This section provides configuration examples for Basic IP Multicast Routing.

Example: Configuring an IP Multicast Boundary

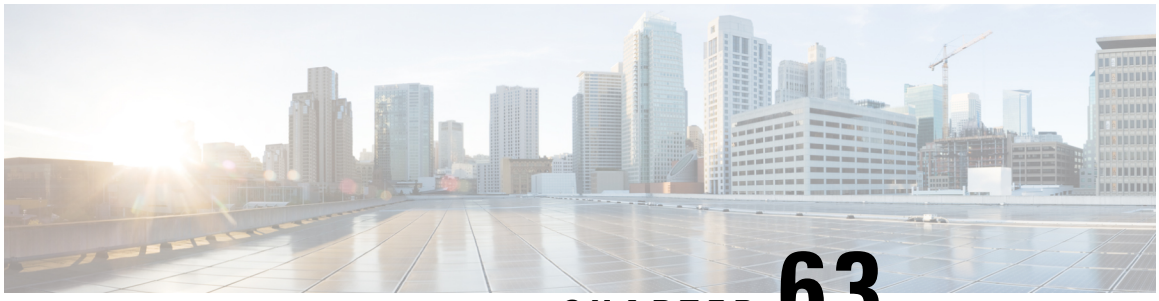
This example shows how to set up a boundary for all administratively-scoped addresses:

```
(config)# access-list 1 deny 239.0.0.0 0.255.255.255
(config)# access-list 1 permit 224.0.0.0 15.255.255.255
(config)# interface gigabitethernet1/1
(config-if)# ip multicast boundary 1
```

Example: Responding to mrinfo Requests

The software answers mrinfo requests sent by mrouted systems and Cisco routers and multilayer switches. The software returns information about neighbors through DVMRP tunnels and all the routed interfaces. This information includes the metric (always set to 1), the configured TTL threshold, the status of the interface, and various flags. You can also use the **mrinfo** privileged EXEC command to query the router or switch itself, as in this example:

```
Switch# mrinfo
  171.69.214.27 (mm1-7kd.cisco.com) [version cisco 11.1] [flags: PMS]:
  171.69.214.27 -> 171.69.214.26 (mm1-r7kb.cisco.com) [1/0/pim/querier]
  171.69.214.27 -> 171.69.214.25 (mm1-45a.cisco.com) [1/0/pim/querier]
  171.69.214.33 -> 171.69.214.34 (mm1-45c.cisco.com) [1/0/pim]
  171.69.214.137 -> 0.0.0.0 [1/0/pim/querier/down/leaf]
  171.69.214.203 -> 0.0.0.0 [1/0/pim/querier/down/leaf]
  171.69.214.18 -> 171.69.214.20 (mm1-45e.cisco.com) [1/0/pim]
  171.69.214.18 -> 171.69.214.19 (mm1-45c.cisco.com) [1/0/pim]
  171.69.214.18 -> 171.69.214.17 (mm1-45a.cisco.com) [1/0/pim]
```



CHAPTER 63

Configuring Multicast Routing over GRE Tunnel

Prerequisites for Configuring Multicast Routing over GRE Tunnel

Before configuring multicast routing over GRE, you should be familiar with the concepts of IP Multicast Routing Technology and GRE Tunneling.

Restrictions for Configuring Multicast Routing over GRE Tunnel

The following are the restrictions for configuring multicast routing over GRE tunnel:

- IPv6 multicast over GRE tunnel is not supported.
- The total number of supported multicast routes (mroutes) is 1024, across all tunnels.
- Bidirectional PIM is not supported.
- Multicast routing should be configured on the first hop router (FHR), the rendezvous point (RP) and the last hop router (LHR) to support multicast over the GRE tunnel.
- The tunnel source can be a loopback, physical, or L3 EtherChannel interface.
- No feature interactions such as IPSec, ACL, Tunnel counters, Crypto support, Fragmentation, Cisco Discovery Protocol (CDP), QoS, GRE keepalive, Multipoint GRE, etc. are supported on the GRE Tunnel.
- Tunnel source cannot be a subinterface.

Information About Multicast Routing over GRE Tunnel

This chapter describes how to configure a Generic Route Encapsulation (GRE) tunnel to tunnel IP multicast packets between non-IP multicast areas. The benefit is that IP multicast traffic can be sent from a source to a multicast group, over an area where IP multicast is not supported. Multicast Routing over GRE Tunnel supports

sparse mode and pim-ssm mode; and supports static RP and auto-RP. See Rendevous Point and Auto-RP for information on configuring static RP and auto-RP.

Benefits of Tunneling to Connect Non-IP Multicast Areas

If the path between a source and a group member (destination) does not support IP multicast, a tunnel between them can transport IP multicast packets.

How to Configure Multicast Routing over GRE Tunnel

This section provides steps for configuring multicast routing over GRE tunnel.

Configuring a GRE Tunnel to Connect Non-IP Multicast Areas

You can configure a GRE tunnel to transport IP multicast packets between a source and destination that are connected by a medium that does not support multicast routing.

Procedure

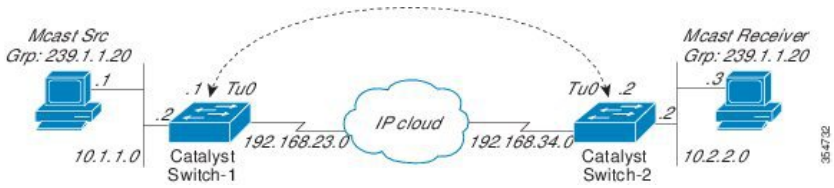
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip multicast-routing** **Example:** `Device(config)# ip multicast-routing`	Enables IP multicast routing.
Step 4	**interface tunnel** *number* **Example:** `Device(config)# interface tunnel 0`	Enters tunnel interface configuration mode.
Step 5	**ip address ip_address** *subnet_mask* **Example:** `Device(config-if)# ip address 192.168.24.1 255.255.255.252`	Configures IP address and IP subnet.

	Command or Action	Purpose
Step 6	**ip pim sparse-mode** **Example:** `Device(config-if)# ip pim sparse-mode`	Enables sparse mode of operation of Protocol Independent Multicast (PIM) on the tunnel interface with one of the following mode of operation:
Step 7	**tunnel source** { *ip-address* | *interface-name* } **Example:** `Device(config-if)# tunnel source 100.1.1.1`	Configures the tunnel source.
Step 8	**tunnel destination** { *hostname* | *ip-address* } **Example:** `Device(config-if)# tunnel destination 100.1.5.3`	Configures the tunnel destination.
Step 9	**end** **Example:** `Device(config-if)# end`	Ends the current configuration session and returns to privileged EXEC mode.
Step 10	**show interface** *type number* **Example:** `Device# show interface tunnel 0`	Displays tunnel interface information.

Tunneling to Connect Non-IP Multicast Areas Example

The following example shows multicast-routing between a switch through a GRE tunnel.

Figure 69: Tunnel Connecting Non-IP Multicast Areas



In the figure above, the multicast source (10.1.1.1) is connected to Switch-1 and is configured for multicast group 239.1.1.20. The multicast receiver (10.2.2.3) is connected to Switch-2 and is configured to receive multicast packets for group 239.1.1.20. Separating Switch-1 and Switch-2 is an IP cloud, which is not configured for multicast routing.

A GRE tunnel is configured between Switch-1 to Switch-2 sourced with their loopback interfaces. Multicast-routing is enabled on Switch-1 and Switch-2. The **ip pim sparse-mode** command is configured on

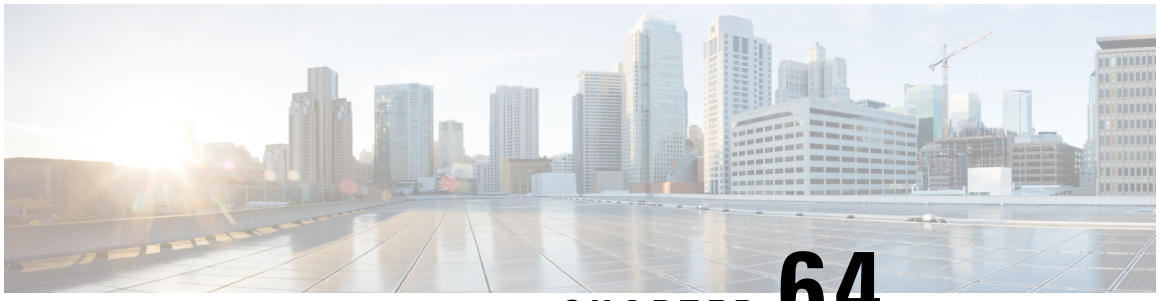
tunnel interfaces to support PIM in the sparse mode. Sparse mode configuration on the tunnel interfaces allows sparse-mode packets to be forwarded over the tunnel depending on rendezvous point (RP) configuration for the group.

Switch-1 Configuration:

```
Device(config)# ip multicast-routing
Device(config)# interface Loopback0   //Tunnel source interface
Device(config-if)# ip address 2.2.2.2 255.255.255.255

Device(config)# interface Tunnel 10     //Tunnel interface configured for PIM
traffic
Device(config-if)# ip address 192.168.24.1 255.255.255.252
Device(config-if)# ip pim sparse-mode
Device(config-if)# ip nhrp map 192.168.24.3  4.4.4.4  //NHRP may optionally be
configured to dynamically discover tunnel end points.
Device(config-if)# ip nhrp map multicast 4.4.4.4
Device(config-if)# ip nhrp network-id 1
Device(config-if)# ip nhrp nhs 192.168.24.3
Device(config-if)# tunnel source Loopback0
Device(config-if)# tunnel destination 4.4.4.4

Device(config)# interface GigabitEthernet 1/1       //Source interface
Device(config-if)# ip address 10.1.1.2 255.255.255.0
Device(config-if)# ip pim sparse-mode
```

Switch-2 Configuration:

```
Device(config)# ip multicast-routing
Device(config)# interface Loopback0    //Tunnel source interface
Device(config-if)# ip address 4.4.4.4 255.255.255.255

Device(config)# interface Tunnel 10      //Tunnel interface configured for PIM
traffic
Device(config-if)# ip address 192.168.24.2 255.255.255.252
Device(config-if)# ip nhrp map 192.168.24.4 2.2.2.2  //NHRP may optionally be
configured to dynamically discover tunnel end points.
Device(config-if)# ip nhrp map multicast 2.2.2.2
Device(config-if)# ip nhrp network-id 1
Device(config-if)# ip nhrp nhs 192.168.24.4
Device(config-if)# ip pim sparse-mode
Device(config-if)# tunnel source Loopback0
Device(config-if)# tunnel destination 2.2.2.2

Device(config)# interface GigabitEthernet 1/2      //Receiver interface
Device(config-if)# ip address 10.2.2.2 255.255.255.0
Device(config-if)# ip pim sparse-mode
```



CHAPTER 64

Configuring IGMP

Prerequisites for IGMP and IGMP Snooping

Prerequisites for IGMP Snooping

Observe these guidelines when configuring the IGMP snooping querier:

- Configure the VLAN in global configuration mode.
- Configure an IP address on the VLAN interface. When enabled, the IGMP snooping querier uses the IP address as the query source address.
- If there is no IP address configured on the VLAN interface, the IGMP snooping querier tries to use the configured global IP address for the IGMP querier. If there is no global IP address specified, the IGMP querier tries to use the VLAN device virtual interface (SVI) IP address (if one exists). If there is no SVI IP address, the device uses the first available IP address configured on the device. The first IP address available appears in the output of the **show ip interface** privileged EXEC command. The IGMP snooping querier does not generate an IGMP general query if it cannot find an available IP address on the device.
- The IGMP snooping querier supports IGMP Versions 1 and 2.
- When administratively enabled, the IGMP snooping querier moves to the nonquerier state if it detects the presence of a multicast router in the network.
- When it is administratively enabled, the IGMP snooping querier moves to the operationally disabled state under these conditions:
    - IGMP snooping is disabled in the VLAN.

- PIM is enabled on the SVI of the corresponding VLAN.

Restrictions for IGMP and IGMP Snooping

Restrictions for Configuring IGMP

The following are the restrictions for configuring IGMP:

- The device supports IGMP Versions 1, 2 , and 3.

Note For IGMP Version 3, only IGMP Version 3 BISS (Basic IGMPv3 Snooping Support) is supported.

- IGMP Version 3 uses new membership report messages that might not be correctly recognized by older IGMP snooping devices.
- IGMPv3 can operate with both ISM and SSM. In ISM, both exclude and include mode reports are applicable. In SSM, only include mode reports are accepted by the last-hop router. Exclude mode reports are ignored.
- Use ACLs to designate a specified port only as a multicast host port and not as a multicast router port. Multicast router control-packets received on this port are dropped.

Restrictions for IGMP Snooping

The following are the restrictions for IGMP snooping:

- The device supports IGMPv3 snooping based only on the destination multicast IP address. It does not support snooping based on a source IP address or proxy report.
- IGMPv3 join and leave messages are not supported on the devices running IGMP filtering or Multicast VLAN registration (MVR).
- IGMP report suppression is supported only when the multicast query has IGMPv1 and IGMPv2 reports. This feature is not supported when the query includes IGMPv3 reports.
- The IGMP configurable leave time is only supported on hosts running IGMP Version 2. IGMP version 2 is the default version for the device.

  The actual leave latency in the network is usually the configured leave time. However, the leave time might vary around the configured time, depending on real-time CPU load conditions, network delays and the amount of traffic sent through the interface.
- The IGMP throttling action restriction can be applied only to Layer 2 ports. You can use **ip igmp max-groups action replace** interface configuration command on a logical EtherChannel interface but cannot use it on ports that belong to an EtherChannel port group.

  When the maximum group limitation is set to the default (no maximum), entering the **ip igmp max-groups action** {**deny** | **replace**} command has no effect.

If you configure the throttling action and set the maximum group limitation after an interface has added multicast entries to the forwarding table, the forwarding-table entries are either aged out or removed, depending on the throttling action.

Information about IGMP

Role of the Internet Group Management Protocol

IGMP is used to dynamically register individual hosts in a multicast group on a particular LAN. Enabling PIM on an interface also enables IGMP. IGMP provides a means to automatically control and limit the flow of multicast traffic throughout your network with the use of special multicast queriers and hosts.

- A querier is a network device, such as a router, that sends query messages to discover which network devices are members of a given multicast group.
- A host is a receiver, including routers, that sends report messages (in response to query messages) to inform the querier of a host membership. Hosts use IGMP messages to join and leave multicast groups.

Hosts identify group memberships by sending IGMP messages to their local multicast device. Under IGMP, devices listen to IGMP messages and periodically send out queries to discover which groups are active or inactive on a particular subnet.

IGMP Multicast Addresses

IP multicast traffic uses group addresses, which are Class D IP addresses. The high-order four bits of a Class D address are 1110. Therefore, host group addresses can be in the range 224.0.0.0 to 239.255.255.255.

Multicast addresses in the range 224.0.0.0 to 224.0.0.255 are reserved for use by routing protocols and other network control traffic. The address 224.0.0.0 is guaranteed not to be assigned to any group.

IGMP packets are transmitted using IP multicast group addresses as follows:

- IGMP general queries are destined to the address 224.0.0.1 (all systems on a subnet).
- IGMP group-specific queries are destined to the group IP address for which the device is querying.
- IGMP group membership reports are destined to the group IP address for which the device is reporting.
- IGMPv2 leave-group messages are destined to the address 224.0.0.2 (all devices on a subnet).
- IGMPv3 membership reports are destined to the address 224.0.0.22; all IGMPv3-capable multicast devices must listen to this address.

IGMP Versions

The device supports IGMP version 1, IGMP version 2, and IGMP version 3. These versions are interoperable on the device. For example, if IGMP snooping is enabled and the querier's version is IGMPv2, and the device receives an IGMPv3 report from a host, then the device can forward the IGMPv3 report to the multicast router.

An IGMPv3 device can receive messages from and forward messages to a device running the Source Specific Multicast (SSM) feature.

IGMP Version 1

IGMP version 1 (IGMPv1) primarily uses a query-response model that enables the multicast router and multilayer device to find which multicast groups are active (have one or more hosts interested in a multicast group) on the local subnet. IGMPv1 has other processes that enable a host to join and leave a multicast group. For more information, see RFC 1112.

IGMP Version 2

IGMPv2 extends IGMP functionality by providing such features as the IGMP leave process to reduce leave latency, group-specific queries, and an explicit maximum query response time. IGMPv2 also adds the capability for routers to elect the IGMP querier without depending on the multicast protocol to perform this task. For more information, see RFC 2236.

Note IGMP version 2 is the default version for the device.

IGMP Version 3

The device supports IGMP version 3.

An IGMPv3 device supports Basic IGMPv3 Snooping Support (BISS), which includes support for the snooping features on IGMPv1 and IGMPv2 switches and for IGMPv3 membership report messages. BISS constrains the flooding of multicast traffic when your network includes IGMPv3 hosts. It constrains traffic to approximately the same set of ports as the IGMP snooping feature on IGMPv2 or IGMPv1 hosts.

An IGMPv3 device can receive messages from and forward messages to a device running the Source Specific Multicast (SSM) feature.

IGMPv3 Host Signaling

IGMPv3 is the third version of the IETF standards track protocol in which hosts signal membership to last-hop devices of multicast groups. IGMPv3 introduces the ability for hosts to signal group membership that allows filtering capabilities with respect to sources. A host can signal either that it wants to receive traffic from all sources sending to a group except for some specific sources (a mode called EXCLUDE) or that it wants to receive traffic only from some specific sources sending to the group (a mode called INCLUDE).

IGMPv3 can operate with both ISM and SSM. In ISM, both EXCLUDE and INCLUDE mode reports are accepted by the last-hop router. In SSM, only INCLUDE mode reports are accepted by the last-hop router.

IGMP Versions Differences

There are three versions of IGMP, as defined by Request for Comments (RFC) documents of the Internet Engineering Task Force (IETF). IGMPv2 improves over IGMPv1 by adding the ability for a host to signal desire to leave a multicast group and IGMPv3 improves over IGMPv2 mainly by adding the ability to listen to multicast originating from a set of source IP addresses only.

Table 66: IGMP Versions

IGMP Version	Description
IGMPv1	Provides the basic query-response mechanism that allows the multicast device to determine which multicast groups are active and other processes that enable hosts to join and leave a multicast group. RFC 1112 defines the IGMPv1 host extensions for IP multicasting.
IGMPv2	Extends IGMP, allowing such capabilities as the IGMP leave process, group-specific queries, and an explicit maximum response time field. IGMPv2 also adds the capability for devices to elect the IGMP querier without dependence on the multicast protocol to perform this task. RFC 2236 defines IGMPv2.
IGMPv3	Provides for source filtering, which enables a multicast receiver host to signal to a device which groups it wants to receive multicast traffic from, and from which sources this traffic is expected. In addition, IGMPv3 supports the link local address 224.0.0.22, which is the destination IP address for IGMPv3 membership reports; all IGMPv3-capable multicast devices must listen to this address. RFC 3376 defines IGMPv3.

Note By default, enabling a PIM on an interface enables IGMPv2 on that device. IGMPv2 was designed to be as backward compatible with IGMPv1 as possible. To accomplish this backward compatibility, RFC 2236 defined special interoperability rules. If your network contains legacy IGMPv1 hosts, you should be familiar with these operability rules. For more information about IGMPv1 and IGMPv2 interoperability, see RFC 2236, Internet Group Management Protocol, Version 2 .

Devices That Run IGMPv1

IGMPv1 devices send IGMP queries to the “all-hosts” multicast address of 224.0.0.1 to solicit multicast groups with active multicast receivers. The multicast receivers also can send IGMP reports to the device to notify it that they are interested in receiving a particular multicast stream. Hosts can send the report asynchronously or in response to the IGMP queries sent by the device. If more than one multicast receiver exists for the same multicast group, only one of these hosts sends an IGMP report message; the other hosts suppress their report messages.

In IGMPv1, there is no election of an IGMP querier. If more than one device on the segment exists, all the devices send periodic IGMP queries. IGMPv1 has no special mechanism by which the hosts can leave the group. If the hosts are no longer interested in receiving multicast packets for a particular group, they simply do not reply to the IGMP query packets sent from the device. The device continues sending query packets. If the device does not hear a response in three IGMP queries, the group times out and the device stops sending multicast packets on the segment for the group. If the host later wants to receive multicast packets after the timeout period, the host simply sends a new IGMP join to the device, and the device begins to forward the multicast packet again.

If there are multiple devices on a LAN, a designated router (DR) must be elected to avoid duplicating multicast traffic for connected hosts. PIM devices follow an election process to select a DR. The PIM device with the highest IP address becomes the DR.

The DR is responsible for the following tasks:

- Sending PIM register and PIM Join and Prune messages toward the rendezvous point (RP) to inform it about host group membership.
- Sending IGMP host-query messages.
- Sending host-query messages by default every 60 seconds in order to keep the IGMP overhead on hosts and networks very low.

Devices That Run IGMPv2

IGMPv2 improves the query messaging capabilities of IGMPv1.

The query and membership report messages in IGMPv2 are identical to the IGMPv1 messages with two exceptions:

- IGMPv2 query messages are broken into two categories: general queries (identical to IGMPv1 queries) and group-specific queries.
- IGMPv1 membership reports and IGMPv2 membership reports have different IGMP type codes.

IGMPv2 also enhances IGMP by providing support for the following capabilities:

- Querier election process--Provides the capability for IGMPv2 devices to elect the IGMP querier without having to rely on the multicast routing protocol to perform the process.
- Maximum Response Time field--A new field in query messages permits the IGMP querier to specify the maximum query-response time. This field permits the tuning of the query-response process to control response burstiness and to fine-tune leave latencies.
- Group-Specific Query messages--Permits the IGMP querier to perform the query operation on a specific group instead of all groups.
- Leave-Group messages--Provides hosts with a method of notifying devices on the network that they wish to leave the group.

Unlike IGMPv1, in which the DR and the IGMP querier are typically the same device, in IGMPv2 the two functions are decoupled. The DR and the IGMP querier are selected based on different criteria and may be different devices on the same subnet. The DR is the device with the highest IP address on the subnet, whereas the IGMP querier is the device with the lowest IP address.

Query messages are used to elect the IGMP querier as follows:

1. When IGMPv2 devices start, they each multicast a general query message to the all-systems group address of 224.0.0.1 with their interface address in the source IP address field of the message.
2. When an IGMPv2 device receives a general query message, the device compares the source IP address in the message with its own interface address. The device with the lowest IP address on the subnet is elected the IGMP querier.
3. All devices (excluding the querier) start the query timer, which is reset whenever a general query message is received from the IGMP querier. If the query timer expires, it is assumed that the IGMP querier has gone down, and the election process is performed again to elect a new IGMP querier.

By default, the timer is two times the query interval.

Devices Running IGMPv3

IGMPv3 adds support for source filtering, which enables a multicast receiver host to signal to a device which groups it wants to receive multicast traffic from, and from which sources this traffic is expected. This membership information enables the software to forward traffic only from those sources from which receivers requested the traffic.

IGMPv3 supports applications that explicitly signal sources from which they want to receive traffic. With IGMPv3, receivers signal membership to a multicast group in the following two modes:

- INCLUDE mode--In this mode, the receiver announces membership to a group and provides a list of IP addresses (the INCLUDE list) from which it wants to receive traffic.
- EXCLUDE mode--In this mode, the receiver announces membership to a group and provides a list of IP addresses (the EXCLUDE list) from which it does not want to receive traffic. In other words, the host wants to receive traffic only from sources whose IP addresses are not listed in the EXCLUDE list. To receive traffic from all sources, like in the case of the Internet Standard Multicast (ISM) service model, a host expresses EXCLUDE mode membership with an empty EXCLUDE list.

IGMPv3 is the industry-designated standard protocol for hosts to signal channel subscriptions in an SSM network environment. For SSM to rely on IGMPv3, IGMPv3 must be available in the network stack portion of the operating systems running on the last hop devices and hosts and be used by the applications running on those hosts.

In IGMPv3, hosts send their membership reports to 224.0.0.22; all IGMPv3 devices, therefore, must listen to this address. Hosts, however, do not listen or respond to 224.0.0.22; they only send their reports to that address. In addition, in IGMPv3, there is no membership report suppression because IGMPv3 hosts do not listen to the reports sent by other hosts. Therefore, when a general query is sent out, all hosts on the wire respond.

IGMP Join and Leave Process

IGMP Join Process

When a host wants to join a multicast group, the host sends one or more unsolicited membership reports for the multicast group it wants to join. The IGMP join process is the same for IGMPv1 and IGMPv2 hosts.

In IGMPv3, the join process for hosts proceeds as follows:

- When a hosts wants to join a group, it sends an IGMPv3 membership report to 224.0.0.22 with an empty EXCLUDE list.
- When a host wants to join a specific channel, it sends an IGMPv3 membership report to 224.0.0.22 with the address of the specific source included in the INCLUDE list.
- When a host wants to join a group excluding particular sources, it sends an IGMPv3 membership report to 224.0.0.22 excluding those sources in the EXCLUDE list.

Note If some IGMPv3 hosts on a LAN wish to exclude a source and others wish to include the source, then the device will send traffic for the source on the LAN (that is, inclusion trumps exclusion in this situation).

IGMP Leave Process

The method that hosts use to leave a group varies depending on the version of IGMP in operation.

IGMPv1 Leave Process

There is no leave-group message in IGMPv1 to notify the devices on the subnet that a host no longer wants to receive the multicast traffic from a specific group. The host simply stops processing traffic for the multicast group and ceases responding to IGMP queries with IGMP membership reports for the group. As a result, the only way IGMPv1 devices know that there are no longer any active receivers for a particular multicast group on a subnet is when the devices stop receiving membership reports. To facilitate this process, IGMPv1 devices associate a countdown timer with an IGMP group on a subnet. When a membership report is received for the group on the subnet, the timer is reset. For IGMPv1 devices, this timeout interval is typically three times the query interval (3 minutes). This timeout interval means that the device may continue to forward multicast traffic onto the subnet for up to 3 minutes after all hosts have left the multicast group.

IGMPv2 Leave Process

IGMPv2 incorporates a leave-group message that provides the means for a host to indicate that it wishes to stop receiving multicast traffic for a specific group. When an IGMPv2 host leaves a multicast group, if it was the last host to respond to a query with a membership report for that group, it sends a leave-group message to the all-devices multicast group (224.0.0.2).

IGMPv3 Leave Process

IGMPv3 enhances the leave process by introducing the capability for a host to stop receiving traffic from a particular group, source, or channel in IGMP by including or excluding sources, groups, or channels in IGMPv3 membership reports.

IGMP Snooping

Layer 2 can use IGMP snooping to constrain the flooding of multicast traffic by dynamically configuring Layer 2 interfaces so that multicast traffic is forwarded to only those interfaces associated with IP multicast devices. As the name implies, IGMP snooping requires the LAN device to snoop on the IGMP transmissions between the host and the router and to keep track of multicast groups and member ports. When the receives an IGMP report from a host for a particular multicast group, the device adds the host port number to the forwarding table entry; when it receives an IGMP Leave Group message from a host, it removes the host port from the table entry. It also periodically deletes entries if it does not receive IGMP membership reports from the multicast clients.

Note For more information on IP multicast and IGMP, see RFC 1112 and RFC 2236.

The multicast router set on the active device sends out periodic general queries to all VLANs. All hosts interested in this multicast traffic send join requests and are added to the forwarding table entry. The device creates one entry per VLAN in the IGMP snooping IP multicast forwarding table for each group from which it receives an IGMP join request.

The device supports IP multicast group-based bridging, instead of MAC-addressed based groups. With multicast MAC address-based groups, if an IP address being configured translates (aliases) to a previously

configured MAC address or to any reserved multicast MAC addresses (in the range 224.0.0.xxx), the command fails. Because the device uses IP multicast groups, there are no address aliasing issues.

The IP multicast groups learned through IGMP snooping are dynamic. However, you can statically configure multicast groups by using the **ip igmp snooping vlan** *vlan-id* **static** *ip_address* **interface** *interface-id* global configuration command. If you specify group membership for a multicast group address statically, your setting supersedes any automatic manipulation by IGMP snooping. Multicast group membership lists can consist of both user-defined and IGMP snooping-learned settings.

You can configure an IGMP snooping querier to support IGMP snooping in subnets without multicast interfaces because the multicast traffic does not need to be routed.

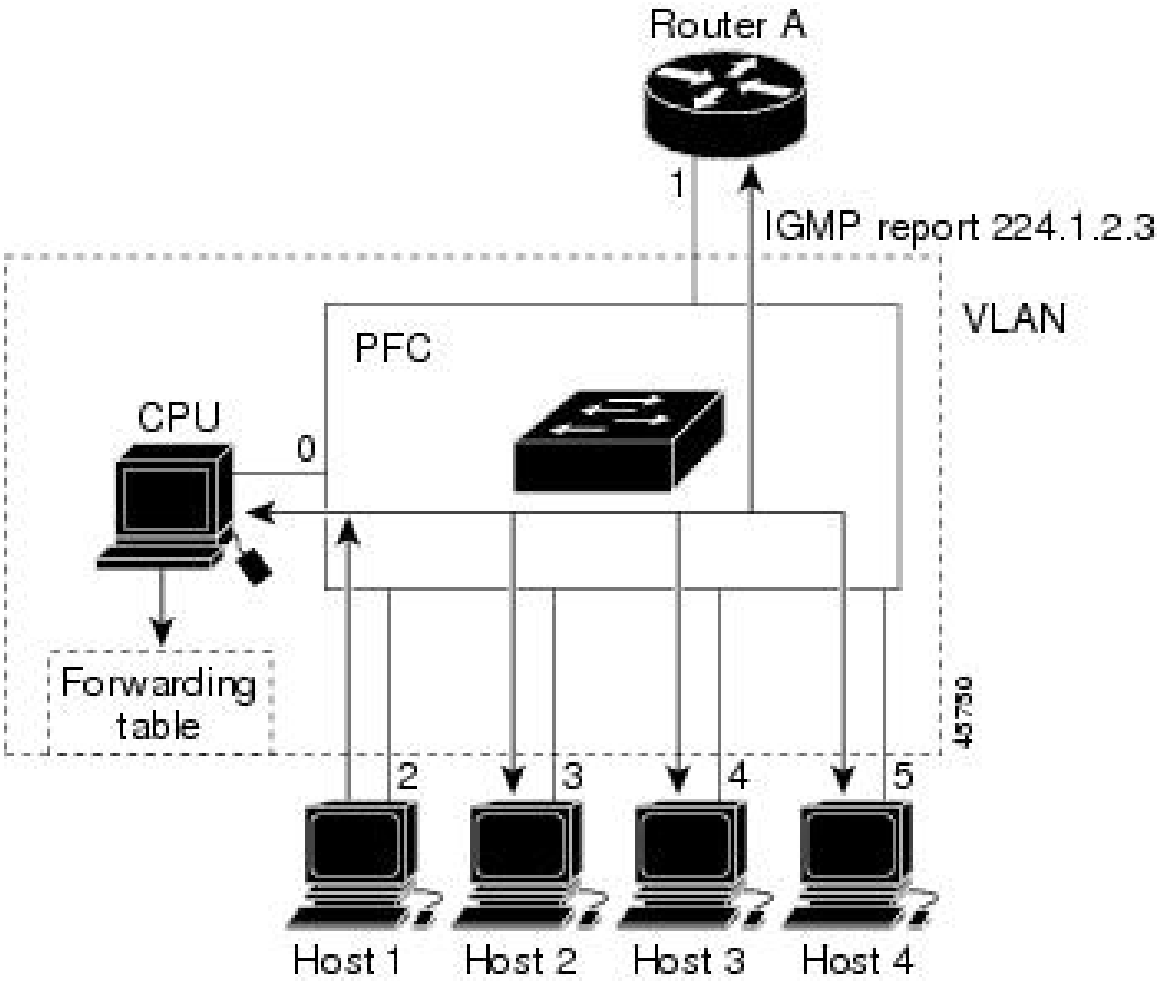
If a port spanning-tree, a port group, or a VLAN ID change occurs, the IGMP snooping-learned multicast groups from this port on the VLAN are deleted.

These sections describe IGMP snooping characteristics:

Joining a Multicast Group

Figure 70: Initial IGMP Join Message

When a host connected to the device wants to join an IP multicast group and it is an IGMP version 2 client, it sends an unsolicited IGMP join message, specifying the IP multicast group to join. Alternatively, when the device receives a general query from the router, it forwards the query to all ports in the VLAN. IGMP version 1 or version 2 hosts wanting to join the multicast group respond by sending a join message to the device. The device CPU creates a multicast forwarding-table entry for the group if it is not already present. The CPU also adds the interface where the join message was received to the forwarding-table entry. The host associated with that interface receives multicast traffic for that multicast group.



Router A sends a general query to the device, which forwards the query to ports 2 through 5, all of which are members of the same VLAN. Host 1 wants to join multicast group 224.1.2.3 and multicasts an IGMP membership report (IGMP join message) to the group. The device CPU uses the information in the IGMP report to set up a forwarding-table entry that includes the port numbers connected to Host 1 and to the router.

Table 67: IGMP Snooping Forwarding Table

Destination Address	Type of Packet	Ports
224.1.2.3	IGMP	1, 2

The device hardware can distinguish IGMP information packets from other packets for the multicast group. The information in the table tells the switching engine to send frames addressed to the 224.1.2.3 multicast IP address that are not IGMP packets to the router and to the host that has joined the group.

Figure 71: Second Host Joining a Multicast Group

If another host (for example, Host 4) sends an unsolicited IGMP join message for the same group, the CPU receives that message and adds the port number of Host 4 to the forwarding table. Because the forwarding table directs IGMP messages only to the CPU, the message is not flooded to other ports on the device. Any

known multicast traffic is forwarded to the group and not to the CPU.

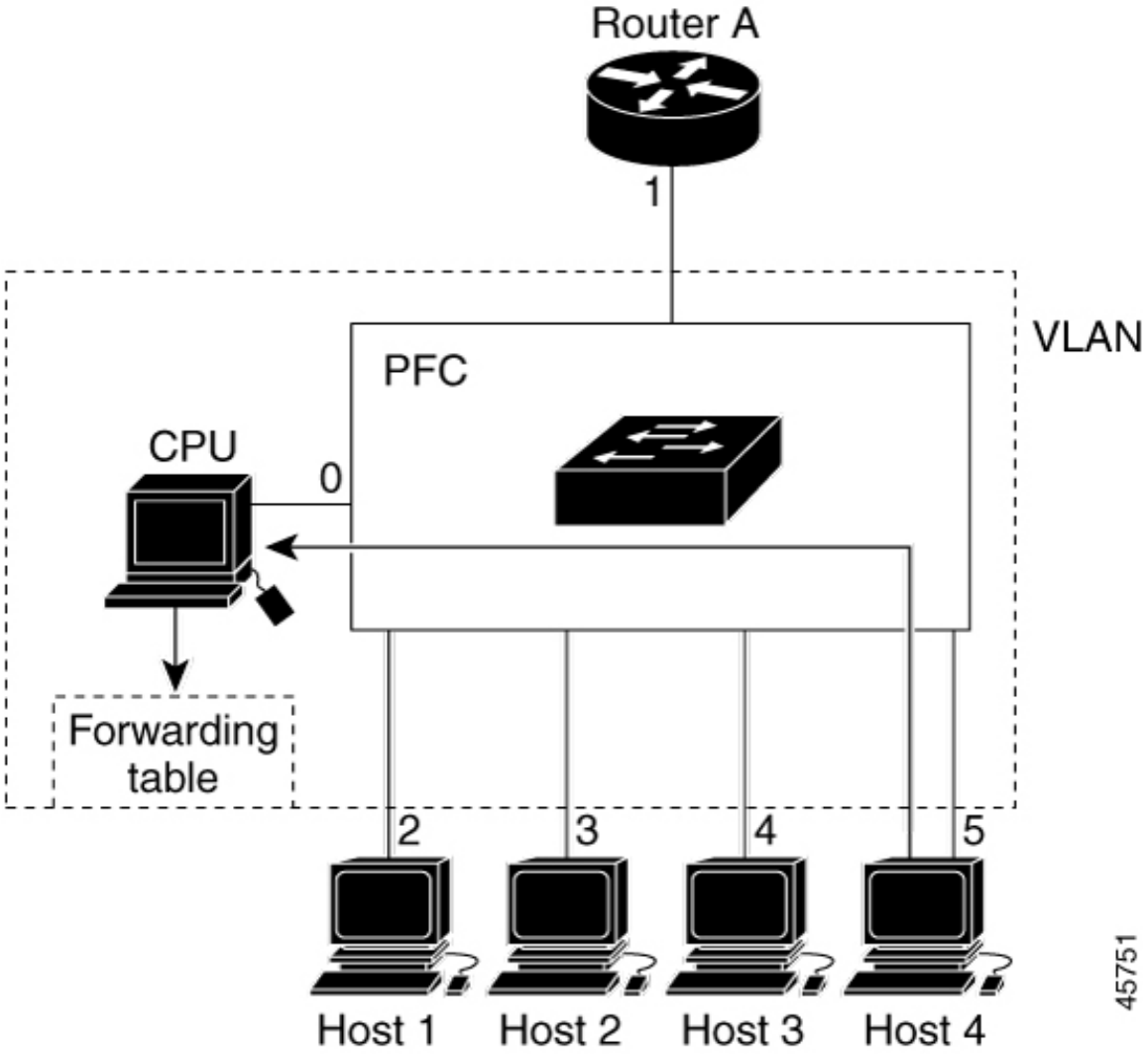


Table 68: Updated IGMP Snooping Forwarding Table

Destination Address	Type of Packet	Ports
224.1.2.3	IGMP	1, 2, 5

Leaving a Multicast Group

The router sends periodic multicast general queries, and the device forwards these queries through all ports in the VLAN. Interested hosts respond to the queries. If at least one host in the VLAN wants to receive multicast traffic, the router continues forwarding the multicast traffic to the VLAN. The device forwards multicast group traffic only to those hosts listed in the forwarding table for that IP multicast group maintained by IGMP snooping.

When hosts want to leave a multicast group, they can silently leave, or they can send a leave message. When the device receives a leave message from a host, it sends a group-specific query to learn if any other devices

connected to that interface are interested in traffic for the specific multicast group. The device then updates the forwarding table for that MAC group so that only those hosts interested in receiving multicast traffic for the group are listed in the forwarding table. If the router receives no reports from a VLAN, it removes the group for the VLAN from its IGMP cache.

Immediate Leave

The device uses IGMP snooping Immediate Leave to remove from the forwarding table an interface that sends a leave message without the device sending group-specific queries to the interface. The VLAN interface is pruned from the multicast tree for the multicast group specified in the original leave message. Immediate Leave ensures optimal bandwidth management for all hosts on a switched network, even when multiple multicast groups are simultaneously in use.

Immediate Leave is only supported on IGMP version 2 hosts. IGMP version 2 is the default version for the device.

Note You should use the Immediate Leave feature only on VLANs where a single host is connected to each port. If Immediate Leave is enabled on VLANs where more than one host is connected to a port, some hosts may be dropped inadvertently.

IGMP Configurable-Leave Timer

You can configure the time that the device waits after sending a group-specific query to determine if hosts are still interested in a specific multicast group. The IGMP leave response time can be configured from 100 to 32767 milliseconds.

IGMP Report Suppression

IGMP report suppression is supported only when the multicast query has IGMPv1 and IGMPv2 reports. This feature is not supported when the query includes IGMPv3 reports.

The device uses IGMP report suppression to forward only one IGMP report per multicast router query to multicast devices. When IGMP report suppression is enabled (the default), the device sends the first IGMP report from all hosts for a group to all the multicast routers. The device does not send the remaining IGMP reports for the group to the multicast routers. This feature prevents duplicate reports from being sent to the multicast devices.

If the multicast router query includes requests only for IGMPv1 and IGMPv2 reports, the device forwards only the first IGMPv1 or IGMPv2 report from all hosts for a group to all the multicast routers.

If the multicast router query also includes requests for IGMPv3 reports, the device forwards all IGMPv1, IGMPv2, and IGMPv3 reports for a group to the multicast devices.

If you disable IGMP report suppression, all IGMP reports are forwarded to the multicast routers.

IGMP Filtering and Throttling

In some environments, for example, metropolitan or multiple-dwelling unit (MDU) installations, you might want to control the set of multicast groups to which a user on a switch port can belong. You can control the distribution of multicast services, such as IP/TV, based on some type of subscription or service plan. You might also want to limit the number of multicast groups to which a user on a switch port can belong.

With the IGMP filtering feature, you can filter multicast joins on a per-port basis by configuring IP multicast profiles and associating them with individual switch ports. An IGMP profile can contain one or more multicast groups and specifies whether access to the group is permitted or denied. If an IGMP profile denying access to a multicast group is applied to a switch port, the IGMP join report requesting the stream of IP multicast traffic is dropped, and the port is not allowed to receive IP multicast traffic from that group. If the filtering action permits access to the multicast group, the IGMP report from the port is forwarded for normal processing. You can also set the maximum number of IGMP groups that a Layer 2 interface can join.

IGMP filtering controls only group-specific query and membership reports, including join and leave reports. It does not control general IGMP queries. IGMP filtering has no relationship with the function that directs the forwarding of IP multicast traffic. The filtering feature operates in the same manner whether CGMP or MVR is used to forward the multicast traffic.

IGMP filtering applies only to the dynamic learning of IP multicast group addresses, not static configuration.

With the IGMP throttling feature, you can set the maximum number of IGMP groups that a Layer 2 interface can join. If the maximum number of IGMP groups is set, the IGMP snooping forwarding table contains the maximum number of entries, and the interface receives an IGMP join report, you can configure an interface to drop the IGMP report or to replace the randomly selected multicast entry with the received IGMP report.

Note IGMPv3 join and leave messages are not supported on a device running IGMP filtering.

Default IGMP Configuration

This table displays the default IGMP configuration for the device.

Table 69: Default IGMP Configuration

Feature	Default Setting
Multilayer device as a member of a multicast group	No group memberships are defined.
Access to multicast groups	All groups are allowed on an interface.
IGMP version	Version 2 on all interfaces.
IGMP host-query message interval	60 seconds on all interfaces.
IGMP query timeout	60 seconds on all interfaces.
IGMP maximum query response time	10 seconds on all interfaces.
Multilayer device as a statically connected member	Disabled.

Default IGMP Snooping Configuration

This table displays the default IGMP snooping configuration for the device.

Table 70: Default IGMP Snooping Configuration

Feature	Default Setting
IGMP snooping	Enabled globally and per VLAN
Multicast routers	None configured
IGMP snooping Immediate Leave	Disabled
Static groups	None configured
TCN[6] flood query count	2
TCN query solicitation	Disabled
IGMP snooping querier	Disabled
IGMP report suppression	Enabled

[6] (1) TCN = Topology Change Notification

Default IGMP Filtering and Throttling Configuration

This table displays the default IGMP filtering and throttling configuration for the device.

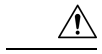
Table 71: Default IGMP Filtering Configuration

Feature	Default Setting
IGMP filters	None applied.
IGMP maximum number of IGMP groups	No maximum set. **Note** When the maximum number of groups is in the forwarding table, the default IGMP throttling action is to deny the IGMP report.
IGMP profiles	None defined.
IGMP profile action	Deny the range addresses.

How to Configure IGMP

Configuring the Device as a Member of a Group

You can configure the device as a member of a multicast group and discover multicast reachability in a network. If all the multicast-capable routers and multilayer devices that you administer are members of a multicast group, pinging that group causes all of these devices to respond. The devices respond to ICMP echo-request packets addressed to a group of which they are members. Another example is the multicast trace-route tools provided in the software.



Caution Performing this procedure might impact the CPU performance because the CPU will receive all data traffic for the group address.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies the interface on which you want to enable multicast routing, and enters interface configuration mode.
Step 4	**ip igmp join-group** *group-address* **Example:** `Device(config-if)# ip igmp join-group 225.2.2.2`	Configures the device to join a multicast group. By default, no group memberships are defined. For *group-address*, specify the multicast IP address in dotted decimal notation.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show ip igmp interface** [*interface-id*] **Example:** `Device# show ip igmp interface GigabitEthernet 1/1`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Changing the IGMP Version

By default, the switch uses IGMP Version 2, which provides features such as the IGMP query timeout and the maximum query response time.

All systems on the subnet must support the same version. The switch does not automatically detect Version 1 systems and switch to Version 1. You can mix Version 1 and Version 2 hosts on the subnet because Version 2 routers or switches always work correctly with IGMPv1 hosts.

Configure the switch for Version 1 if your hosts do not support Version 2.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be configured, and enters the interface configuration mode.
Step 4	**ip igmp version** {**1** | **2** | **3** } **Example:**	Specifies the IGMP version that the switch uses. **Note**

	Command or Action	Purpose
	`Device(config-if)# ip igmp version 2`	If you change to Version 1, you cannot configure the **ip igmp query-interval** or the **ip igmp query-max-response-time** interface configuration commands. To return to the default setting, use the **no ip igmp version** interface configuration command.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show ip igmp interface** [*interface-id*] **Example:** `Device# show ip igmp interface`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Modifying the IGMP Host-Query Message Interval

The device periodically sends IGMP host-query messages to discover which multicast groups are present on attached networks. These messages are sent to the all-hosts multicast group (224.0.0.1) with a time-to-live (TTL) of 1. The device sends host-query messages to refresh its knowledge of memberships present on the network. If, after some number of queries, the software discovers that no local hosts are members of a multicast group, the software stops forwarding multicast packets to the local network from remote origins for that group and sends a prune message upstream toward the source.

The device elects a PIM designated router (DR) for the LAN (subnet). The designated router is responsible for sending IGMP host-query messages to all hosts on the LAN. In sparse mode, the designated router also sends PIM register and PIM join messages toward the RP router. With IGMPv2, the DR is the router or multilayer device with the highest IP address. With IGMPv1, the DR is elected according to the multicast routing protocol that runs on the LAN.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface on which you want to enable multicast routing, and enters interface configuration mode.
Step 4	**ip igmp query-interval** *seconds* **Example:** `Device(config-if)# ip igmp query-interval 75`	Configures the frequency at which the designated router sends IGMP host-query messages. By default, the designated router sends IGMP host-query messages every 60 seconds to keep the IGMP overhead very low on hosts and networks.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show ip igmp interface** [*interface-id*] **Example:** `Device# show ip igmp interface`	Displays
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Changing the Maximum Query Response Time for IGMPv2

If you are using IGMPv2, you can change the maximum query response time advertised in IGMP queries. The maximum query response time enables the device to quickly detect that there are no more directly connected group members on a LAN. Decreasing the value enables the device to prune groups faster.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies the interface on which you want to enable multicast routing, and enters interface configuration mode.
Step 4	**ip igmp query-max-response-time** *seconds* **Example:** `Device(config-if)# ip igmp query-max-response-time 15`	Changes the maximum query response time advertised in IGMP queries. The default is 10 seconds. The range is 1 to 25.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show ip igmp interface** [*interface-id*] **Example:** `Device# show ip igmp interface`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config`	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`startup-config`	

Configuring the Device as a Statically Connected Member

At various times, either there is not a group member on a network segment or a host that cannot report its group membership by using IGMP. However, you may want multicast traffic to be sent to that network segment. The following commands are used to pull multicast traffic down to a network segment:

- **ip igmp join-group**—The device accepts the multicast packets in addition to forwarding them. Accepting the multicast packets prevents the device from fast switching.
- **ip igmp static-group**—The device does not accept the packets itself, but only forwards them. This method enables fast switching. The outgoing interface appears in the IGMP cache, but the device itself is not a member, as evidenced by lack of an L (local) flag in the multicast route entry.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies the interface on which you want to enable multicast routing, and enters interface configuration mode.
Step 4	**ip igmp static-group** *group-address* **Example:** `Device(config-if)# ip igmp static-group 239.100.100.101`	Configures the device as a statically connected member of a group. By default, this feature is disabled.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show ip igmp interface** [*interface-id*] **Example:** `Device# show ip igmp interface GigabitEthernet 1/1`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring IGMP Profiles

Follow these steps to create an IGMP profile:

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp profile** *profile number* **Example:** `Device(config)# ip igmp profile 3`	Assigns a number to the profile you are configuring, and enters IGMP profile configuration mode. The profile number range is 1 to 4294967295. When you are in IGMP profile configuration mode, you can create the profile by using these commands: • **deny**—Specifies that matching addresses are denied; this is the default. • **exit**—Exits from igmp-profile configuration mode. • **no**—Negates a command or returns to its defaults.

	Command or Action	Purpose
		• **permit**—Specifies that matching addresses are permitted. • **range**—Specifies a range of IP addresses for the profile. You can enter a single IP address or a range with a start and an end address. The default for the device is to have no IGMP profiles configured. **Note** To delete a profile, use the **no ip igmp profile** *profile number* global configuration command.
Step 4	**permit** | **deny** **Example:** `Device(config-igmp-profile)# permit`	(Optional) Sets the action to permit or deny access to the IP multicast address. If no action is configured, the default for the profile is to deny access.
Step 5	**range** *ip multicast address* **Example:** `Device(config-igmp-profile)# range 229.9.9.0`	Enters the IP multicast address or range of IP multicast addresses to which access is being controlled. If entering a range, enter the low IP multicast address, a space, and the high IP multicast address. You can use the **range** command multiple times to enter multiple addresses or ranges of addresses. **Note** To delete an IP multicast address or range of IP multicast addresses, use the **no range** *ip multicast address* IGMP profile configuration command.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 7	**show ip igmp profile** *profile number* **Example:** `Device# show ip igmp profile 3`	Verifies the profile configuration.
Step 8	**show running-config** **Example:**	Verifies your entries.

	Command or Action	Purpose
	`Device# show running-config`	
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Applying IGMP Profiles

To control access as defined in an IGMP profile, you have to apply the profile to the appropriate interfaces. You can apply IGMP profiles only to Layer 2 access ports; you cannot apply IGMP profiles to routed ports or SVIs. You cannot apply profiles to ports that belong to an EtherChannel port group. You can apply a profile to multiple interfaces, but each interface can have only one profile applied to it.

Follow these steps to apply an IGMP profile to a switch port:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies the physical interface, and enters interface configuration mode. The interface must be a Layer 2 port that does not belong to an EtherChannel port group.
Step 4	**ip igmp filter** *profile number* **Example:** `Device(config-if)# ip igmp filter 321`	Applies the specified IGMP profile to the interface. The range is 1 to 4294967295. **Note** To remove a profile from an interface, use the **no ip igmp filter** *profile number* interface configuration command.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Setting the Maximum Number of IGMP Groups

Follow these steps to set the maximum number of IGMP groups that a Layer 2 interface can join:

Before you begin

This restriction can be applied to Layer 2 ports only; you cannot set a maximum number of IGMP groups on routed ports or SVIs. You also can use this command on a logical EtherChannel interface but cannot use it on ports that belong to an EtherChannel port group.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/2`	Specifies the interface to be configured, and enters interface configuration mode. The interface can be a Layer 2 port that does not belong to an EtherChannel group or a EtherChannel interface.

	Command or Action	Purpose
Step 4	**ip igmp max-groups** *number* **Example:** `Device(config-if)# ip igmp max-groups 20`	Sets the maximum number of IGMP groups that the interface can join. The range is 0 to 4294967294. The default is to have no maximum set.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config interface** *interface-id* **Example:** `Device# show running-config interface gigabitethernet1/1`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring the IGMP Throttling Action

After you set the maximum number of IGMP groups that a Layer 2 interface can join, you can configure an interface to replace the existing group with the new group for which the IGMP report was received.

Follow these steps to configure the throttling action when the maximum number of entries is in the forwarding table:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the physical interface to be configured, and enters interface configuration mode. The interface can be a Layer 2 port that does not belong to an EtherChannel group or an EtherChannel interface. The interface cannot be a trunk port.
Step 4	**ip igmp max-groups action** {**deny** | **replace**} **Example:** `Device(config-if)# ip igmp max-groups action replace`	When an interface receives an IGMP report and the maximum number of entries is in the forwarding table, specifies the action that the interface takes: • **deny**—Drops the report. If you configure this throttling action, the entries that were previously in the forwarding table are not removed but are aged out. After these entries are aged out and the maximum number of entries is in the forwarding table, the device drops the next IGMP report received on the interface. • **replace**—Replaces the existing group with the new group for which the IGMP report was received. If you configure this throttling action, the entries that were previously in the forwarding table are removed. When the maximum number of entries is in the forwarding table, the device replaces a randomly selected entry with the received IGMP report. To prevent the device from removing the forwarding-table entries, you can configure the IGMP throttling action before an interface adds entries to the forwarding table. **Note** To return to the default action of dropping the report, use the **no ip igmp max-groups action** interface configuration command.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config interface** *interface-id* **Example:**	Verifies your entries.

	Command or Action	Purpose
	`Device# show running-config interface gigabitethernet1/1`	
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring the Device to Forward Multicast Traffic in the Absence of Directly Connected IGMP Hosts

Perform this optional task to configure the device to forward multicast traffic in the absence of directly connected IGMP hosts.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode. • For the *type* and *number* arguments, specify an interface that is connected to hosts.
Step 4	Do one of the following: • **ip igmp join-group** *group-address* • **ip igmp static-group** {***** | *group-address* [**source** *source-address*]} **Example:** `Device(config-if)# ip igmp join-group 225.2.2.2` **Example:**	The first sample shows how to configure an interface on the device to join the specified group. With this method, the device accepts the multicast packets in addition to forwarding them. Accepting the multicast packets prevents the device from fast switching. The second example shows how to configure static group membership entries on an interface.

	Command or Action	Purpose
	`Device(config-if)# ip igmp static-group 225.2.2.2`	With this method, the device does not accept the packets itself, but only forwards them. Hence, this method allows fast switching. The outgoing interface appears in the IGMP cache, but the device itself is not a member, as evidenced by lack of an "L" (local) flag in the multicast route entry
Step 5	**end** **Example:** `Device#(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show ip igmp interface** [*interface-type interface-number*] **Example:** `Device# show ip igmp interface`	(Optional) Displays multicast-related information about an interface.

Controlling Access to an SSM Network Using IGMP Extended Access Lists

Perform this optional task to control access to an SSM network by using an IGMP extended access list that filters SSM traffic based on source address, group address, or both.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip multicast-routing** [**distributed**] **Example:** `Device(config)# ip multicast-routing distributed`	Enables IP multicast routing. • The **distributed** keyword is required for IPv4 multicast..
Step 4	**ip pim ssm** {**default** | **range** *access-list*} **Example:**	Configures SSM service. • The **default** keyword defines the SSM range access list as 232/8.

	Command or Action	Purpose
	`Device(config)# ip pim ssm default`	• The **range** keyword specifies the standard IP access list number or name that defines the SSM range.
Step 5	**ip access-list extended** *access-list* -name **Example:** `Device(config)# ip access-list extended mygroup`	Specifies an extended named IP access list.
Step 6	**deny igmp** *source source-wildcard destination destination-wildcard* [*igmp-type*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**] **Example:** `Device(config-ext-nacl)# deny igmp host 10.1.2.3 any`	(Optional) Filters the specified source address or group address from the IGMP report, thereby restricting hosts on a subnet from membership to the (S, G) channel. • Repeat this step to restrict hosts on a subnet membership to other (S, G) channels. (These sources should be more specific than a subsequent **permit** statement because any sources or groups not specifically permitted are denied.) • Remember that the access list ends in an implicit **deny** statement. • This example shows how to create a deny statement that filters all groups for source 10.1.2.3, which effectively denies the source.
Step 7	**permit igmp** *source source-wildcard destination destination-wildcard* [*igmp-type*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**] **Example:** `Device(config-ext-nacl)# permit igmp any any`	Allows a source address or group address in an IGMP report to pass the IP access list. • You must have at least one **permit** statement in an access list. • Repeat this step to allow other sources to pass the IP access list. • This example shows how to allow group membership to sources and groups not denied by prior **deny** statements.
Step 8	**exit** **Example:** `Device(config-ext-nacl)# exit`	Exits the current configuration session and returns to global configuration mode.
Step 9	interface type number **Example:**	Selects an interface that is connected to hosts on which IGMPv3 can be enabled.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet 1/1`	
Step 10	**ip igmp access-group** *access-list* **Example:** `Device(config-if)# ip igmp access-group mygroup`	Applies the specified access list to IGMP reports.
Step 11	**ip pim sparse-mode** **Example:** `Device(config-if)# ip pim sparse-mode`	Enables PIM-SM on the interface. **Note** You must use sparse mode.
Step 12	Repeat Steps 1 through 11 on all interfaces that require access control of SSM channel membership.	--
Step 13	**ip igmp version 3** **Example:** `Device(config-if)# ip igmp version 3`	Enables IGMPv3 on this interface. The default version of IGMP is IGMP version 2. Version 3 is required by SSM.
Step 14	Repeat Step 13 on all host-facing interfaces.	--
Step 15	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

How to Configure IGMP Snooping

Enabling IGMP Snooping

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ip igmp snooping** **Example:** `Device(config)# ip igmp snooping`	Globally enables IGMP snooping after it has been disabled.
Step 4	**bridge-domain** *bridge-id* **Example:** `Device(config)# bridge-domain 100`	(Optional) Enters bridge domain configuration mode.
Step 5	**ip igmp snooping** **Example:** `Device(config-bdomain)# ip igmp snooping`	(Optional) Enables IGMP snooping on the bridge domain interface being configured. • Required only if IGMP snooping was previously explicitly disabled on the specified bridge domain.
Step 6	**end** **Example:** `Device(config-bdomain)# end`	Returns to privileged EXEC mode.

Enabling or Disabling IGMP Snooping on a VLAN Interface

Follow these steps to enable IGMP snooping on a VLAN interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp snooping vlan** *vlan-id* **Example:** `Device(config)# ip igmp snooping vlan 7`	Enables IGMP snooping on the VLAN interface. The VLAN ID range is 1 to 1001 and 1006 to 4094. IGMP snooping must be globally enabled before you can enable VLAN snooping.

	Command or Action	Purpose
		Note To disable IGMP snooping on a VLAN interface, use the **no ip igmp snooping vlan** *vlan-id* global configuration command for the specified VLAN number.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Setting the Snooping Method

Multicast-capable router ports are added to the forwarding table for every Layer 2 multicast entry. The device learns of the ports through one of these methods:

- Snooping on IGMP queries, Protocol-Independent Multicast (PIM) packets
- Statically connecting to a multicast router port using the **ip igmp snooping mrouter** global configuration command

Beginning in privileged EXEC mode, follow these steps to alter the method in which a VLAN interface accesses a multicast router:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip igmp snooping vlan** *vlan-id* **mrouter interface** {**GigabitEthernet** | **Port-Channel** } **Example:** `Device(config)# ip igmp snooping vlan 1 mrouter interface GigabitEthernet1/3`	Enables IGMP snooping on a VLAN. The VLAN ID range is 1 to 1001 and 1006 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show ip igmp snooping** **Example:** `Device# show ip igmp snooping`	Verifies the configuration.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Multicast Router Port

Perform these steps to add a multicast router port (enable a static connection to a multicast router) on the device.

Note Static connections to multicast routers are supported only on device ports.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ip igmp snooping vlan** *vlan-id* **mrouter interface** *interface-id* **Example:** `Device(config)# ip igmp snooping vlan 5 mrouter interface GigabitEthernet 1/1`	Specifies the multicast router VLAN ID and the interface to the multicast router. • The VLAN ID range is 1 to 1001 and 1006 to 4094. • The interface can be a physical interface or a port channel. The port-channel range is 1 to 128. **Note** To remove a multicast router port from the VLAN, use the **no ip igmp snooping vlan** *vlan-id* **mrouter interface** *interface-id* global configuration command.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 5	**show ip igmp snooping mrouter** [**vlan** *vlan-id*] **Example:** `Device# show ip igmp snooping mrouter vlan 5`	Verifies that IGMP snooping is enabled on the VLAN interface.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Host Statically to Join a Group

Hosts or Layer 2 ports normally join multicast groups dynamically, but you can also statically configure a host on an interface.

Follow these steps to add a Layer 2 port as a member of a multicast group:

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enabled privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp snooping vlan** *vlan-id* **static** *ip_address* **interface** *interface-id* **Example:** `Device(config)# ip igmp snooping vlan 105 static 230.0.0.1 interface gigabitethernet1/1`	Statically configures a Layer 2 port as a member of a multicast group: • *vlan-id* is the multicast group VLAN ID. The range is 1 to 1001 and 1006 to 4094. • *ip-address* is the group IP address. • *interface-id* is the member port. It can be a physical interface or a port channel (1 to 128). **Note** To remove the Layer 2 port from the multicast group, use the **no ip igmp snooping vlan** *vlan-id* **static** *mac-address* **interface** *interface-id* global configuration command.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 5	**show ip igmp snooping groups** **Example:** `Device# show ip igmp snooping groups`	Verifies the member port and the IP address.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Enabling IGMP Immediate Leave

When you enable IGMP Immediate Leave, the device immediately removes a port when it detects an IGMP Version 2 leave message on that port. You should use the Immediate-Leave feature only when there is a single receiver present on every port in the VLAN.

Note Immediate Leave is supported only on IGMP Version 2 hosts. IGMP Version 2 is the default version for the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp snooping vlan** *vlan-id* **immediate-leave** **Example:** `Device(config)# ip igmp snooping vlan 21 immediate-leave`	Enables IGMP Immediate Leave on the VLAN interface. **Note** To disable IGMP Immediate Leave on a VLAN, use the **no ip igmp snooping vlan** *vlan-id* **immediate-leave** global configuration command.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show ip igmp snooping vlan** *vlan-id* **Example:** `Device# show ip igmp snooping vlan 21`	Verifies that Immediate Leave is enabled on the VLAN interface.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the IGMP Leave Timer

You can configure the leave time globally or on a per-VLAN basis. Follow these steps to enable the IGMP configurable-leave timer:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp snooping last-member-query-interval** *time* **Example:** `Device(config)# ip igmp snooping last-member-query-interval 1000`	Configures the IGMP leave timer globally. The range is 100 to 32767 milliseconds. The default leave time is 1000 milliseconds. **Note** To globally reset the IGMP leave timer to the default setting, use the **no ip igmp snooping last-member-query-interval** global configuration command.
Step 4	**ip igmp snooping vlan** *vlan-id* **last-member-query-interval** *time* **Example:** `Device(config)# ip igmp snooping vlan 210 last-member-query-interval 1000`	(Optional) Configures the IGMP leave time on the VLAN interface. The range is 100 to 32767 milliseconds. **Note** Configuring the leave time on a VLAN overrides the globally configured timer. **Note** To remove the configured IGMP leave-time setting from the specified VLAN, use the **no ip igmp snooping vlan** *vlan-id* **last-member-query-interval** global configuration command.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show ip igmp snooping** **Example:** `Device# show ip igmp snooping`	(Optional) Displays the configured IGMP leave time.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring the IGMP Robustness-Variable

Use the following procedure to configure the IGMP robustness variable on the device.

The robustness variable is the integer used by IGMP snooping during calculations for IGMP messages. The robustness variable provides fine tuning to allow for expected packet loss.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp snooping robustness-variable** *count* **Example:** `Device(config)# ip igmp snooping robustness-variable 3`	Configures the IGMP robustness variable. The range is 1 to 3 times. The recommended value for the robustness variable is 2. Use this command to change the value of the robustness variable for IGMP snooping from the default (2) to a specified value.
Step 4	**ip igmp snooping vlan** *vlan-id* **robustness-variable** *count* **Example:** `Device(config)#ip igmp snooping vlan 100`	(Optional) Configures the IGMP robustness variable on the VLAN interface. The range is 1 to 3 times. The recommended value for the robustness variable is 2. **Note**

	Command or Action	Purpose
	`robustness-variable 3`	Configuring the robustness variable count on a VLAN overrides the globally configured value.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show ip igmp snooping** **Example:** `Device# show ip igmp snooping`	(Optional) Displays the configured IGMP robustness variable count.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring the IGMP Last Member Query Count

To configure the number of times the device sends IGMP group-specific or group-source-specific (with IGMP version 3) query messages in response to receiving a group-specific or group-source-specific leave message, use this command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp snooping last-member-query-count** *count* **Example:** `Device(config)# ip igmp snooping`	Configures the IGMP last member query count. The range is 1 to 7 messages. The default is 2 messages.

	Command or Action	Purpose
	`last-member-query-count 3`	
Step 4	**ip igmp snooping vlan** *vlan-id* **last-member-query-count** *count* **Example:** `Device(config)#ip igmp snooping vlan 100 last-member-query-count 3`	(Optional) Configures the IGMP last member query count on the VLAN interface. The range is 1 to 7 messages. **Note** Configuring the last member query count on a VLAN overrides the globally configured timer.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show ip igmp snooping** **Example:** `Device# show ip igmp snooping`	(Optional) Displays the configured IGMP last member query count.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring TCN-Related Commands

Controlling the Multicast Flooding Time After a TCN Event

You can configure the number of general queries by which multicast data traffic is flooded after a topology change notification (TCN) event. If you set the TCN flood query count to 1 the flooding stops after receiving 1 general query. If you set the count to 7, the flooding continues until 7 general queries are received. Groups are relearned based on the general queries received during the TCN event.

Some examples of TCN events are when the client location is changed and the receiver is on same port that was blocked but is now forwarding, and when a port goes down without sending a leave message.

Follow these steps to configure the TCN flood query count:

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enabled privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp snooping tcn flood query count** *count* **Example:** `Device(config)# ip igmp snooping tcn flood query count 3`	Specifies the number of IGMP general queries for which the multicast traffic is flooded. The range is 1 to 10. The default, the flooding query count is 2. **Note** To return to the default flooding query count, use the **no ip igmp snooping tcn flood query count** global configuration command.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 5	**show ip igmp snooping** **Example:** `Device# show ip igmp snooping`	Verifies the TCN settings.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Recovering from Flood Mode

When a topology change occurs, the spanning-tree root sends a special IGMP leave message (also known as global leave) with the group multicast address 0.0.0.0. However, you can enable the device to send the global leave message whether it is the spanning-tree root or not. When the router receives this special leave, it immediately sends general queries, which expedite the process of recovering from the flood mode during the TCN event. Leaves are always sent if the device is the spanning-tree root regardless of this configuration.

Follow these steps to enable sending of leave messages:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp snooping tcn query solicit** **Example:** `Device(config)# ip igmp snooping tcn query solicit`	Sends an IGMP leave message (global leave) to speed the process of recovering from the flood mode caused during a TCN event. By default, query solicitation is disabled. **Note** To return to the default query solicitation, use the **no ip igmp snooping tcn query solicit** global configuration command.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show ip igmp snooping** **Example:** `Device# show ip igmp snooping`	Verifies the TCN settings.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Disabling Multicast Flooding During a TCN Event

When the device receives a TCN, multicast traffic is flooded to all STP non-edge ports until 2 general queries are received. The switch doesn't not flood multicast traffic to STP edge ports after STP TCN events. If the device has many ports with attached hosts that are subscribed to different multicast groups, this flooding might

exceed the capacity of the link and cause packet loss. You can use the **no** form of **ip igmp snooping tcn flood** interface configuration command to control this behavior.

Follow these steps to disable multicast flooding on an interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies the interface to be configured, and enters interface configuration mode.
Step 4	**no ip igmp snooping tcn flood** **Example:** `Device(config-if)# no ip igmp snooping tcn flood`	Disables the flooding of multicast traffic during a spanning-tree TCN event. By default, multicast flooding is enabled on an interface. **Note** To re-enable multicast flooding on an interface, use the **ip igmp snooping tcn flood** interface configuration command.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 6	**show ip igmp snooping** **Example:** `Device# show ip igmp snooping`	Verifies the TCN settings.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config`	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`startup-config`	

Configuring the IGMP Snooping Querier

Follow these steps to enable the IGMP snooping querier feature in a VLAN:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enabled privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp snooping querier** **Example:** `Device(config)# ip igmp snooping querier`	Enables the IGMP snooping querier.
Step 4	**ip igmp snooping querier address** *ip_address* **Example:** `Device(config)# ip igmp snooping querier address 172.16.24.1`	(Optional) Specifies an IP address for the IGMP snooping querier. If you do not specify an IP address, the querier tries to use the global IP address configured for the IGMP querier. **Note** The IGMP snooping querier does not generate an IGMP general query if it cannot find an IP address on the device.
Step 5	**ip igmp snooping querier query-interval** *interval-count* **Example:** `Device(config)# ip igmp snooping querier query-interval 30`	(Optional) Sets the interval between IGMP queriers. The range is 1 to 18000 seconds.
Step 6	**ip igmp snooping querier tcn query** [**count** *count* | **interval** *interval*] **Example:**	(Optional) Sets the time between Topology Change Notification (TCN) queries. The count range is 1 to 10. The interval range is 1 to 255 seconds.

	Command or Action	Purpose
	`Device(config)# ip igmp snooping querier tcn query interval 20`	
Step 7	**ip igmp snooping querier timer expiry** *timeout* **Example:** `Device(config)# ip igmp snooping querier timer expiry 180`	(Optional) Sets the length of time until the IGMP querier expires. The range is 60 to 300 seconds.
Step 8	**ip igmp snooping querier version** *version* **Example:** `Device(config)# ip igmp snooping querier version 2`	(Optional) Selects the IGMP version number that the querier feature uses. Select 1 or 2.
Step 9	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 10	**show ip igmp snooping vlan** *vlan-id* **Example:** `Device# show ip igmp snooping vlan 30`	(Optional) Verifies that the IGMP snooping querier is enabled on the VLAN interface. The VLAN ID range is 1 to 1001 and 1006 to 4094.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Disabling IGMP Report Suppression

Follow these steps to disable IGMP report suppression:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enabled privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no ip igmp snooping report-suppression** **Example:** `Device(config)# no ip igmp snooping report-suppression`	Disables IGMP report suppression. When report suppression is disabled, all IGMP reports are forwarded to the multicast routers. IGMP report suppression is enabled by default. When IGMP report supression is enabled, the device forwards only one IGMP report per multicast router query. **Note** To re-enable IGMP report suppression, use the **ip igmp snooping report-suppression** global configuration command.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 5	**show ip igmp snooping** **Example:** `Device# show ip igmp snooping`	Verifies that IGMP report suppression is disabled.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring IGMP

You can display specific statistics, such as the contents of IP routing tables, caches, and databases.

Note This release does not support per-route statistics.

You can display information to learn resource usage and solve network problems. You can also display information about node reachability and discover the routing path that packets of your device are taking through the network.

You can use any of the privileged EXEC commands in the following table to display various routing statistics.

Table 72: Commands for Displaying System and Network Statistics

Command	Purpose
show ip igmp filter	Displays IGMP filter information.
show ip igmp groups [*type-number* | *detail*]	Displays the multicast groups that are directly connecte to the device and that were learned through IGMP.
show ip igmp interface [*type number*]	Displays multicast-related information about an interfac
show ip igmp membership [*name/group address* | **all** | **tracked**]	Displays IGMP membership information for forwardin
show ip igmp profile [*profile_number*]	Displays IGMP profile information.
show ip igmp ssm-mapping [*hostname/IP address*]	Displays IGMP SSM mapping information.
show ip igmp static-group {**class-map** [**interface** [*type*]]	Displays static group information.
show ip igmp vrf	Displays the selected VPN routing/forwarding instance name. **Note** The **show ip igmp vrf** *vrf-name* **snooping groups** command ignores the **vrf** keyword and displays the snooping information for the VLANs. Use the **show ip igmp snooping groups** command to see the IGMP snooping information for the VLANs.

Monitoring IGMP Snooping Information

You can display IGMP snooping information for dynamically learned and statically configured router ports and VLAN interfaces. You can also display MAC address multicast entries for a VLAN configured for IGMP snooping.

Table 73: Commands for Displaying IGMP Snooping Information

Command	Purpose
show ip igmp snooping detail	Displays the operational state information.

Command	Purpose
show ip igmp snooping groups [**count** |**dynamic** [**count**] | **user** [**count**]]	Displays multicast table information for the device or about a specific parameter: • **count**—Displays the total number of entries for the specified command options instead of the actual entries. • **dynamic**—Displays entries learned through IGMP snooping. • **user**—Displays only the user-configured multicast entries.
show ip igmp snooping groups [**count** | [**vlan** *vlan-id* [*A.B.C.D* | **count**]]	Displays multicast table information for the device or about a specific parameter: • **count**—Displays the total number of groups. • **vlan**—Displays group information by VLAN ID.
show ip igmp snooping igmpv2-tracking	Displays the IGMP snooping tracking. **Note** This command displays group and IP address entries only for wireless multicast IGMP joins and not for wired IGMP joins. Wireless IP multicast must be enabled for this command to display.
show ip igmp snooping groups vlan *vlan-id* [*ip_address* | **count** | **dynamic** [**count**] | **user**[**count**]]	Displays multicast table information for a multicast VLAN or about a specific parameter for the VLAN: • *vlan-id*—The VLAN ID range is 1 to 1001 and 1006 to 4094. • **count**—Displays the total number of entries for the specified command options instead of the actual entries. • **dynamic**—Displays entries learned through IGMP snooping. • *ip_address*—Displays characteristics of the multicast group with the specified group IP address. • **user**—Displays only the user-configured multicast entries.

Command	Purpose
show ip igmp snooping mrouter [**vlan** *vlan-id*]	Displays information on dynamically learned and manually configured multicast router interfaces. **Note** When you enable IGMP snooping, the device automatically learns the interface to which a multicast router is connected. These are dynamically learned interfaces. (Optional) Enter **vlan** *vlan-id* to display information for a single VLAN.
show ip igmp snooping querier [**detail** | **vlan** *vlan-id*]	Displays information about the IP address and receiving port for the most-recently received IGMP query messages in the VLAN. (Optional) Enter **detail** to display the detailed IGMP querier information in a VLAN. (Optional) Enter **vlan** *vlan-id* to display information for a single VLAN.
show ip igmp snooping querier [**vlan** *vlan-id*] **detail**	Displays information about the IP address and receiving port of the most-recently received IGMP query message in the VLAN and the configuration and operational state of the IGMP snooping querier in the VLAN.
show ip igmp snooping [**vlan** *vlan-id* [**detail**]]	Displays the snooping configuration information for all VLANs on the device or for a specified VLAN. (Optional) Enter **vlan** *vlan-id* to display information for a single VLAN. The VLAN ID range is 1 to 1001 and 1006 to 4094.

Monitoring IGMP Filtering and Throttling Configuration

You can display IGMP profile characteristics, and you can display the IGMP profile and maximum group configuration for all interfaces on the device or for a specified interface. You can also display the IGMP throttling configuration for all interfaces on the device or for a specified interface.

Table 74: Commands for Displaying IGMP Filtering and Throttling Configuration

Command	Purpose
show ip igmp profile [*profile number*]	Displays the specified IGMP profile or all the IGMP profiles defined on the device.

Command	Purpose
show running-config [**interface** *interface-id*]	Displays the configuration of the specified interface or the configuration of all interfaces on the device, including (if configured) the maximum number of IGMP groups to which an interface can belong and the IGMP profile applied to the interface.

Configuration Examples for IGMP

Example: Configuring the Device as a Member of a Multicast Group

This example shows how to enable the device to join multicast group 255.2.2.2:

```
Device(config)# interface gigabitethernet1/1
Device(config-if)# ip igmp join-group 255.2.2.2
Device(config-if)#
```

Example: Controlling Access to Multicast Groups

To limit the number of joins on the interface, configure the port for filter which associates with the IGMP profile.

```
Device# configure terminal
Device(config)# ip igmp profile 10
Device(config-igmp-profile)# ?

IGMP profile configuration commands:
deny matching addresses are denied
exit Exit from igmp profile configuration mode
no Negate a command or set its defaults
permit matching addresses are permitted
range add a range to the set

Device(config-igmp-profile)# range 172.16.5.1
Device(config-igmp-profile)# exit
Device(config)# interface gigabitethernet1/1
Device(config-if)# ip igmp filter 10
```

Examples: Configuring IGMP Snooping

This example shows how to enable a static connection to a multicast router:

```
Device# configure terminal
Device(config)# ip igmp snooping vlan 200 mrouter interface gigabitethernet1/2
Device(config)# end
```

This example shows how to statically configure a host on a port:

```
Device# configure terminal
Device(config)# ip igmp snooping vlan 105 static 224.2.4.12 interface gigabitethernet1/1
Device(config)# end
```

This example shows how to enable IGMP Immediate Leave on VLAN 130:

```
Device# configure terminal
Device(config)# ip igmp snooping vlan 130 immediate-leave
Device(config)# end
```

This example shows how to set the IGMP snooping querier source address to 10.0.0.64:

```
Device# configure terminal
Device(config)# ip igmp snooping querier 10.0.0.64
Device(config)# end
```

This example shows how to set the IGMP snooping querier maximum response time to 25 seconds:

```
Device# configure terminal
Device(config)# ip igmp snooping querier query-interval 25
Device(config)# end
```

This example shows how to set the IGMP snooping querier timeout to 60 seconds:

```
Device# configure terminal
Device(config)# ip igmp snooping querier timer expiry 60
Device(config)# end
```

This example shows how to set the IGMP snooping querier feature to Version 2:

```
Device# configure terminal
Device(config)# no ip igmp snooping querier version 2
Device(config)# end
```

Example: Configuring IGMP Profiles

This example shows how to create IGMP profile 4 allowing access to the single IP multicast address and how to verify the configuration. If the action was to deny (the default), it would not appear in the **show ip igmp profile** output display.

```
Device(config)# ip igmp profile 4
Device(config-igmp-profile)# permit
Device(config-igmp-profile)# range 229.9.9.0
Device(config-igmp-profile)# end
Device# show ip igmp profile 4
IGMP Profile 4
    permit
    range 229.9.9.0 229.9.9.0
```

Example: Applying IGMP Profile

This example shows how to apply IGMP profile 4 to a port:

```
Device(config)# interface gigabitethernet1/2
Device(config-if)# ip igmp filter 4
Device(config-if)# end
```

Example: Setting the Maximum Number of IGMP Groups

This example shows how to limit to 25 the number of IGMP groups that a port can join:

```
Device(config)# interface Gigabitethernet1/2
Device(config-if)# ip igmp max-groups 25
Device(config-if)# end
```

Example: Interface Configuration as a Routed Port

This example shows how to configure an interface on the device as a routed port. This configuration is required on the interface for several IP multicast routing configuration procedures that require running the **no switchport** command.

```
Device# configure terminal
Device(config)#  interface GigabitEthernet1/9
Device(config-if)# description interface to be use as routed port
Device(config-if)# no switchport
Device(config-if)# ip address 10.20.20.1 255.255.255.0
Device(config-if)# ip pim sparse-mode
Device(config-if)# ip igmp join-group 224.1.2.3 source 15.15.15.2
Device(config-if)# end
Device# configure terminal
Device# show run interface gigabitEthernet 1/9

Current configuration : 166 bytes
!
interface GigabitEthernet1/9
 no switchport
 ip address 10.20.20.1 255.255.255.0
 ip pim sparse-mode
 ip igmp static-group 224.1.2.3 source 15.15.15.2
end
```

Example: Interface Configuration as an SVI

This example shows how to configure an interface on the device as an SVI. This configuration is required on the interface for several IP multicast routing configuration procedures that require running the **no switchport** command.

```
Device(config)# interface vlan 150
Device(config-if)# ip address 10.20.20.1 255.255.255.0
Device(config-if)# ip pim sparse-mode
Device(config-if)#  ip igmp join-group 224.1.2.3 source 15.15.15.2
Device(config-if)# end
Device# configure terminal
Device(config)# ip igmp snooping vlan 20 static 224.1.2.3 interface gigabitEthernet 1/9
Device# show run interface vlan 150

Current configuration : 137 bytes
!
interface vlan 150
```

```
 ip address 10.20.20.1 255.255.255.0
 ip pim sparse-mode
 ip igmp static-group 224.1.2.3 source 15.15.15.2
end
```

Example: Configuring the Device to Forward Multicast Traffic in the Absence of Directly Connected IGMP Hosts

The following example shows how to configure a device to forward multicast traffic in the absence of directly connected IGMP hosts using the **ip igmp join-group** command. With this method, the device accepts the multicast packets in addition to forwarding them. Accepting the multicast packets prevents the device from fast switching.

In this example, GigabitEthernet interface 1/1 on the device is configured to join the group 225.2.2.2:

```
interface GigabitEthernet1/1
 ip igmp join-group 225.2.2.2
```

Controlling Access to an SSM Network Using IGMP Extended Access Lists

This section contains the following configuration examples for controlling access to an SSM network using IGMP extended access lists:

Note Keep in mind that access lists are very flexible: there are many combinations of permit and deny statements one could use in an access list to filter multicast traffic. The examples in this section simply provide a few examples of how it can be done.

Example: Denying All States for a Group G

The following example shows how to deny all states for a group G. In this example, GigabitEthernet interface 1/1 is configured to filter all sources for SSM group 232.2.2.2 in IGMPv3 reports, which effectively denies this group.

```
ip access-list extended test1
 deny igmp any host 232.2.2.2
 permit igmp any any
!
interface GigabitEthernet 1/1
 ip igmp access-group test1
```

Example: Denying All States for a Source S

The following example shows how to deny all states for a source S. In this example, Gigabit Ethernet interface 1/1 is configured to filter all groups for source 10.2.1.32 in IGMPv3 reports, which effectively denies this source.

```
ip access-list extended test2
 deny igmp host 10.2.1.32 any
 permit igmp any any
```

```
!
interface GigabitEthernet1/1
 ip igmp access-group test2
```

Example: Permitting All States for a Group G

The following example shows how to permit all states for a group G. In this example, Gigabit Ethernet interface 1/2 is configured to accept all sources for SSM group 232.1.1.10 in IGMPv3 reports, which effectively accepts this group altogether.

```
ip access-list extended test3
 permit igmp any host 232.1.1.10
!
interface GigabitEthernet 1/2
 ip igmp access-group test3
```

Example: Permitting All States for a Source S

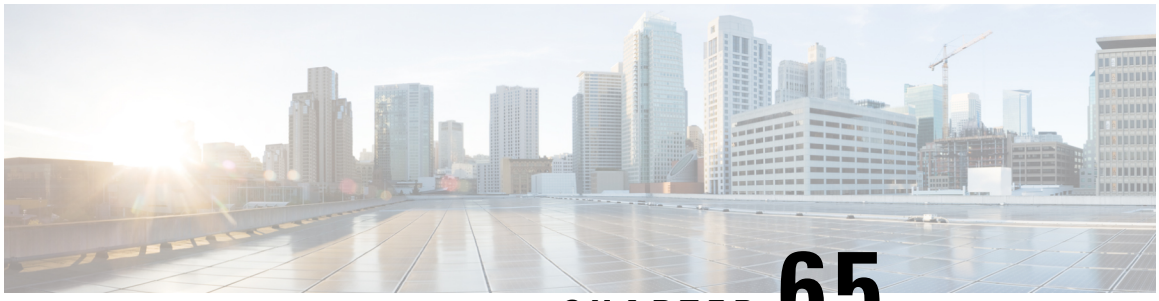
The following example shows how to permit all states for a source S. In this example, Gigabit Ethernet interface 1/2 is configured to accept all groups for source 10.6.23.32 in IGMPv3 reports, which effectively accepts this source altogether.

```
ip access-list extended test4
 permit igmp host 10.6.23.32 any
!
interface GigabitEthernet1/2
 ip igmp access-group test4
```

Example: Filtering a Source S for a Group G

The following example shows how to filter a particular source S for a group G. In this example, Gigabit Ethernet interface 1/2 is configured to filter source 232.2.2.2 for SSM group 232.2.30.30 in IGMPv3 reports.

```
ip access-list extended test5
 deny igmp host 10.4.4.4 host 232.2.30.30
 permit igmp any any
!
interface GigabitEthernet1/2
 ip igmp access-group test5
```



CHAPTER 65

Configuring IGMP Proxy

Prerequisites for IGMP Proxy

- All devices on the IGMP UDL have the same subnet address. If all devices on the UDL cannot have the same subnet address, the upstream device must be configured with secondary addresses to match all of the subnets to which the downstream devices are attached.
- IP multicast is enabled and the PIM interfaces are configured. When you are configuring PIM interfaces for IGMP proxy, use PIM sparse mode (PIM-SM) when the interface is operating in a sparse-mode region and you are running static RP, bootstrap (BSR), or Auto-RP with the Auto-RP listener capability.

Information About IGMP Proxy

IGMP Proxy

An IGMP proxy enables hosts in a unidirectional link routing (UDLR) environment that are not directly connected to a downstream router to join a multicast group sourced from an upstream network.

There are two methods of implementing IGMP Proxy:

- IGMP Proxy for a Single Upstream Interface
- IGMP Proxy for Multiple Upstream Interfaces

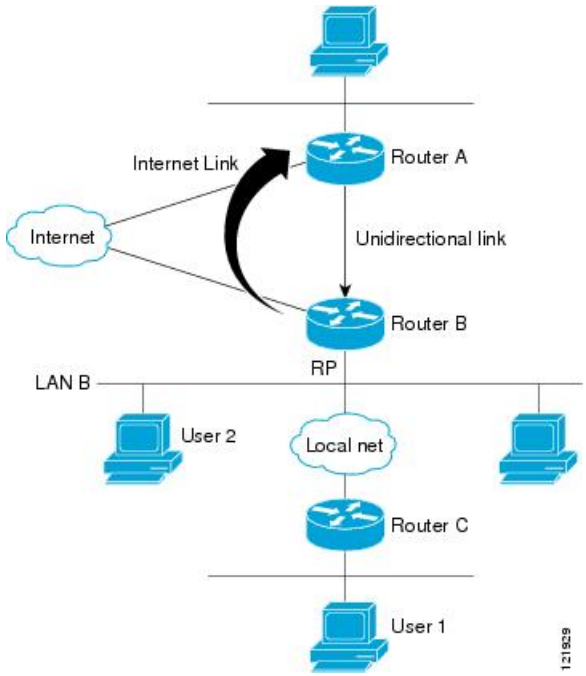
IGMP Proxy for a Single Upstream Interface

The figure below illustrates a sample topology that shows two UDLR scenarios:

- Traditional UDL routing scenario--A UDL device with directly connected receivers.
- IGMP proxy scenario--UDL device without directly connected receivers.

IGMP UDLs are needed on the upstream and downstream devices.

Note Although the following illustration and example uses routers in the configuration, any device (router or switch) can be used.



Scenario 1 - Traditional UDLR Scenario (UDL Device with Directly Connected Receivers)

For scenario 1, no IGMP proxy mechanism is needed. In this scenario, the following sequence of events occurs:

1. User 2 sends an IGMP membership report requesting interest in group G.
2. Router B receives the IGMP membership report, adds a forwarding entry for group G on LAN B, and proxies the IGMP report to Router A, which is the UDLR upstream device.
3. The IGMP report is then proxied across the Internet link.
4. Router A receives the IGMP proxy and maintains a forwarding entry on the unidirectional link.

Scenario 2 - IGMP Proxy Scenario (UDL Device without Directly Connected Receivers)

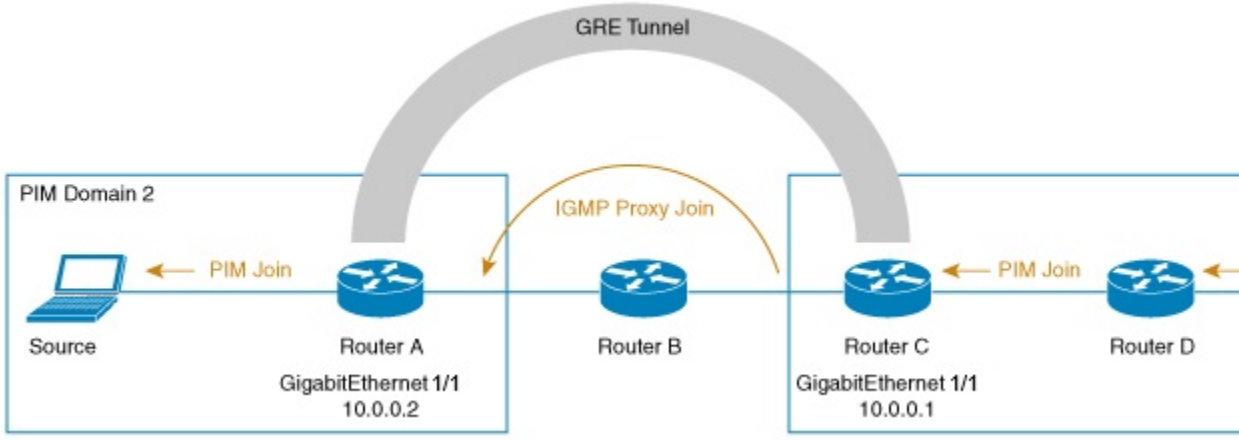
For scenario 2, the IGMP proxy mechanism is needed to enable hosts that are not directly connected to a downstream device to join a multicast group sourced from an upstream network. In this scenario, the following sequence of events occurs:

1. User 1 sends an IGMP membership report requesting interest in group G.
2. Router C sends a PIM Join message hop-by-hop to the RP (Router B).
3. Router B receives the PIM Join message and adds a forwarding entry for group G on LAN B.

4. Router B periodically checks its mroute table and proxies the IGMP membership report to its upstream UDL device across the Internet link.
5. Router A creates and maintains a forwarding entry on the unidirectional link (UDL).

In an enterprise network, it is desirable to be able to receive IP multicast traffic via satellite and forward the traffic throughout the network. With unidirectional link routing (UDLR) alone, scenario 2 would not be possible because receiving hosts must be directly connected to the downstream device, Router B. The IGMP proxy mechanism overcomes this limitation by creating an IGMP report for (*, G) entries in the multicast forwarding table. To make this scenario functional, therefore, you must enable IGMP report forwarding of proxied (*, G) multicast static route (mroute) entries (using the **ip igmp mroute-proxy** command) and enable the mroute proxy service (using the **ip igmp proxy-service** command) on interfaces leading to PIM-enabled networks with potential members.

Note Because PIM messages are not forwarded upstream, each downstream network and the upstream network have a separate domain.



Scenario 3 - IGMP Proxy Scenario without UDLR

For scenario 3, the IGMP proxy mechanism is used to enable hosts to receive traffic from an upstream network, without using a UDLR link. In this scenario, the following sequence of events occurs:

1. The host is in PIM Domain 1 and sends an IGMP membership report (a join request) to ***Router D*** requesting interest in group G. ***Router D*** converts the IGMP Join to a PIM join and sends it to ***Router C***. This request should now be sent upstream, from ***Router C*** to ***Router A***. The routers are in two different PIM domains (not PIM neighbors) and are connected through a GRE tunnel instead.
2. ***Router C*** converts the PIM join message to an IGMP proxy join so that it can be forwarded to the GRE tunnel endpoint, which is ***Router A***.

Note IGMP proxy joins can be transferred across 1 hop only.

In the figure above, the GRE tunnel provides this single hop between Router C and Router A (bypassing Router B).

In the absence of a GRE tunnel, devices from different PIM domains must have directly (back-to-back) connected interfaces.

3. After the IGMP proxy join reaches ***Router A***, it is forwarded to the source device as a PIM join message.

IGMP Proxy for Multiple Upstream Interfaces

IGMP proxy also enables the user to request data from multiple upstream interfaces. You can implement IGMP proxy by this method if there are more number of upstream devices in the network. With this method, you can also implement IGMP proxy for a single upstream device as in any of the three scenarios described in the previous section.

In this method, IGMP proxy is used to enable the user to receive traffic from multiple upstream devices. The following sequence of events occurs:

1. The host is in PIM Domain 1 and sends multiple IGMP membership reports (join requests) to **Router C** requesting interest in different groups. **Router C** converts the IGMP joins to PIM joins and sends them to **Router B**. These requests should be sent upstream, from **Router B** to **Router A**. The routers are in two different PIM domains (not PIM neighbors).
2. **Router B** converts the PIM join messages to IGMP proxy joins so that it can be forwarded to the upward interfaces.
3. A class-map is configured globally. This class-map describes the information about multicast groups. The IGMP proxy joins for different multicast groups are sent if the following conditions are met:
   - There is (*, G) or and (S, G) entry for that group.
   - (*, G) or the (S, G) entry has a NON-NULL OIF list.
4. During the IGMP proxy intervals, the IGMP proxy joins for different groups are sent through the respective upstream interfaces.
5. After the IGMP proxy join reaches **Router A**, it is forwarded to the different source devices as PIM join messages.

How to Configure IGMP Proxy

Configuring the Upstream UDL Device for IGMP UDLR

Perform this task to configure the upstream UDL device for IGMP UDLR.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode. • For the *type* and *number* arguments, specify the interface to be used as the UDL on the upstream device.

	Command or Action	Purpose
Step 4	**ip igmp unidirectional-link** **Example:** `Device(config-if)# ip igmp unidirectional-link`	Configures IGMP on the interface to be unidirectional for IGMP UDLR.
Step 5	**end** **Example:** `Device(config-if)# end`	Ends the current configuration session and returns to privileged EXEC mode.

Configuring the Downstream UDL Device for IGMP UDLR with IGMP Proxy Support

Perform this task to configure the downstream UDL device for IGMP UDLR with IGMP proxy support.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode. • For the *type* and *number* arguments, specify the interface to be used as the UDL on the downstream device for IGMP UDLR.
Step 4	**ip igmp unidirectional-link** **Example:** `Device(config-if)# ip igmp unidirectional-link`	Configures IGMP on the interface to be unidirectional for IGMP UDLR.
Step 5	**exit** **Example:**	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
	`Device(config-if)# exit`	
Step 6	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode. • For the *type* and *number* arguments, select an interface that is facing the nondirectly connected hosts.
Step 7	**ip igmp mroute-proxy** *type number* **Example:** `Device(config-if)# ip igmp mroute-proxy loopback 0`	Enables IGMP report forwarding of proxied (*, G) multicast static route (mroute) entries. • This step is performed to enable the forwarding of IGMP reports to a proxy service interface for all (*, G) forwarding entries in the multicast forwarding table. • In this example, the **ip igmp mroute-proxy** command is configured on Gigabit Ethernet interface 1/1 to request that IGMP reports be sent to loopback interface 0 for all groups in the mroute table that are forwarded to Gigabit Ethernet interface 1/1.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**interface** *type number* **Example:** `Device(config)# interface loopback 0`	Enters interface configuration mode for the specified interface. • In this example, loopback interface 0 is specified.
Step 10	**ip igmp helper-address udl** *interface-type interface-number* **Example:** `Device(config-if)# ip igmp helper-address udl gigabitethernet 1/1`	Configures IGMP helpering for UDLR. • This step allows the downstream device to helper IGMP reports received from hosts to an upstream device connected to a UDL associated with the interface specified for the *interface-type* and *interface-number* arguments. • In the example topology, IGMP helpering is configured over loopback interface 0 on the downstream device. Loopback interface 0, thus, is configured to helper IGMP reports from hosts to an upstream device connected to Gigabit Ethernet interface 1/1.

	Command or Action	Purpose
Step 11	**ip igmp proxy-service** **Example:** `Device(config-if)# ip igmp proxy-service`	Enables the mroute proxy service. • When the mroute proxy service is enabled, the device periodically checks the static mroute table for (*, G) forwarding entries that match interfaces configured with the **ip igmp mroute-proxy** command (see Step 7) based on the IGMP query interval. Where there is a match, one IGMP report is created and received on this interface. **Note** The **ip igmp proxy-service** command is intended to be used with the **ip igmp helper-address** (UDL) command. • In this example, the **ip igmp proxy-service** command is configured on loopback interface 0 to enable the forwarding of IGMP reports out the interface for all groups on interfaces registered through the **ip igmp mroute-proxy** command (see Step 7).
Step 12	**end** **Example:** `Device(config-if)# end`	Ends the current configuration session and returns to privileged EXEC mode.
Step 13	**show ip igmp interface** **Example:** `Device# show ip igmp interface`	(Optional) Displays multicast-related information about an interface.
Step 14	**show ip igmp udlr** **Example:** `Device# show ip igmp udlr`	(Optional) Displays UDLR information for directly connected multicast groups on interfaces that have a UDL helper address configured.

Configuring the Downstream Device for IGMP Proxy Join without UDLR

Perform this task to configure the downstream device for IGMP Proxy without UDLR.

(Referring to the figure above, all the steps are configured on ***Router C***)

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device > enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode. For the *type* and *number* arguments, specify the interface that is facing the host.
Step 4	**ip igmp mroute-proxy** *type number* **Example:** `Device(config-if)# ip igmp mroute-proxy loopback 0`	Enables the forwarding of IGMP reports to the specified proxy service interface, for forwarding of all proxied (*, G) multicast static route (mroute) entries in the multicast forwarding table. In the step example, *loopback interface 0* is such a proxy service interface.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 6	**interface** *type number* **Example:** `Device(config)# interface loopback 0`	Enters interface configuration mode for the specified proxy service interface. In the step example, *loopback interface 0* is specified.
Step 7	**ip igmp helper-address** *ip-address* **Example:** `Device(config-if)# ip igmp helper-address 10.0.0.2`	Configures IGMP helper for IGMP proxy join. For the *ip-address* argument, specify the ip address of the upstream device to which the IGMP proxy join should reach. In the example topology, the IGMP helper is configured over loopback interface 0 on the downstream device (Router C). This command configures loopback interface 0 to help convert the PIM joins received from Router D to IGMP proxy joins and transfer them to the upstream device (Router A).
Step 8	**ip igmp proxy-service**	Enables the mroute proxy service.

	Command or Action	Purpose
	Example: `Device(config-if) ip igmp proxy-service`	When the mroute proxy service is enabled, the device periodically checks the static mroute table for (*, G) forwarding entries that match interfaces configured with the **ip igmp mroute-proxy** command (see Step 7) based on the IGMP query interval. Where there is a match, one IGMP report is created and received on this interface. **Note** The **ip igmp proxy-service** command is intended to be used with the **ip igmp helper-address** command.
Step 9	**end** **Example:** `Device(config-if)# end`	Ends the current configuration session and returns to privileged EXEC mode.
Step 10	**show ip igmp interface** **Example:** `Device# show ip igmp interface`	(Optional) Displays multicast-related information about an interface.

Configuring the Downstream Device for IGMP Proxy for Multiple Upstream Interfaces

Perform this task to configure the downstream device for IGMP Proxy for multiple upstream interfaces.

(Referring to the figure earlier, all the steps are configured on the *Router B* interface facing the upstream device)

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device > enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map type multicast-flows** *name* **Example:** `Device(config-if)# class-map type multicast-flows proxymap`	Configures the interface with a class-map where the uplink interfaces for different multicast groups are defined.

	Command or Action	Purpose
		The range of the multicast groups is from group 225.0.0.1 to 225.0.0.10.
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet1/1`	Enters interface configuration mode. For the *type* and *number* arguments, specify the interface that is facing the host.
Step 5	**ip igmp upstream-proxy** *class-map-name* **Example:** `Device(config-if)# ip igmp upstream-proxy proxymap`	Enables the interface with an IGMP proxy. The IGMP proxy joins for those groups in the class map are sent if the following conditions are met: • (*,G) or (S,G) mroute exists for the group in the same mvrf as the interface. • (*,G) or (S,G) mroute has a NON-NULL OIF list.
Step 6	**ip igmp iif-starg** **Example:** `Device(config-if)# ip igmp iif-starg`	Alters the RPF interface of the mroutes of those groups specified in the class map to GigabitEthernet1/1.
Step 7	**ip igmp proxy-report-interval** *time* **Example:** `Device(config-if)# ip igmp proxy-report-interval 130`	Configures the interval (in seconds) at which the proxy reports will be sent. The default value is 60 seconds.
Step 8	**end** **Example:** `Device(config-if)# end`	Ends the current configuration session and returns to privileged EXEC mode.
Step 9	**show ip igmp interface** **Example:** `Device# show ip igmp interface`	(Optional) Displays multicast-related information about an interface.

Configuration Examples for IGMP Proxy

Example: Configuring the Upstream UDL Device for IGMP UDLR

The following example shows how to configure the upstream UDL device for IGMP UDLR:

```
interface gigabitethernet 1/1
ip address 10.1.1.1 255.255.255.0
ip pim sparse-mode
!
interface gigabitethernet 1/2
```

```
ip address 10.2.1.1 255.255.255.0
ip pim sparse-mode
ip igmp unidirectional-link
!
interface gigabitethernet 1/3
ip address 10.3.1.1 255.255.255.0
```

Example: Configuring the Downstream UDL Device for IGMP UDLR with IGMP Proxy Support

The following example shows how to configure the downstream UDL device for IGMP UDLR with IGMP proxy support:

```
ip pim rp-address 10.5.1.1 5
access-list 5 permit 239.0.0.0 0.255.255.255
!
interface loopback 0
ip address 10.7.1.1 255.255.255.0
ip pim sparse-mode
ip igmp helper-address udl ethernet 0
ip igmp proxy-service
!
interface gigabitethernet 1/1
ip address 10.2.1.2 255.255.255.0
ip pim sparse-mode
ip igmp unidirectional-link
!
interface gigabitethernet 1/2
ip address 10.5.1.1 255.255.255.0
ip pim sparse-mode
ip igmp mroute-proxy loopback 0
!
interface gigabitethernet 1/3
ip address 10.6.1.1 255.255.255.0
```

Example: Configuring the Downstream Device for IGMP Proxy Join without UDLR

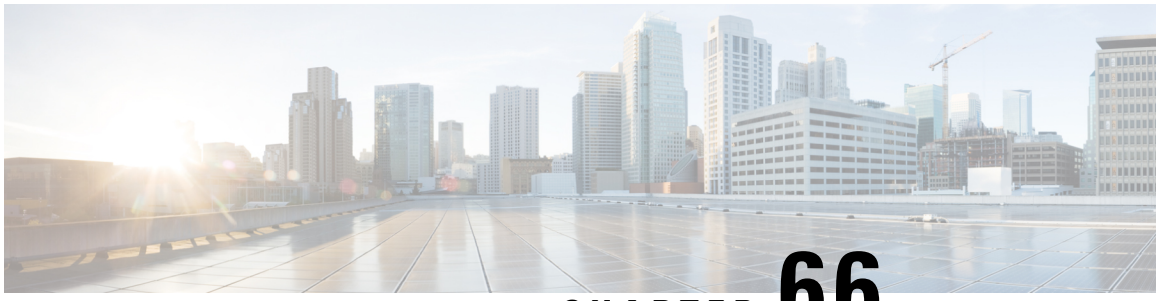
The following example shows how to configure the downstream device for IGMP proxy without UDLR:

```
interface Loopback0
ip address 2.2.2.2 255.255.0.0
ip pim sparse-dense-mode
ip igmp helper-address 99.99.99.1
ip igmp proxy-service
ip ospf 1 area 0
!
```

Example: Configuring the Downstream Device for IGMP Proxy for Multiple Upstream Interfaces

The following example shows how to configure the downstream device for IGMP proxy for multiple upstream interfaces.

```
interface gigabitethernet1/1
ip address 99.99.99.1 255.255.255.0
ip pim passive
ip igmp upstream-proxy l2
ip igmp iif-starg
ip igmp proxy-report-interval 100
end

class-map type multicast-flows l2
group 229.0.0.1
group 228.0.0.1 to 228.0.0.10
```



CHAPTER 66

Constraining IP Multicast in Switched Ethernet

Prerequisites for Constraining IP Multicast in a Switched Ethernet Network

Before using the tasks in this module, you should be familiar with the concepts described in the IP Multicast Routing Technology Overview, on page 795 module.

Information About IP Multicast in a Switched Ethernet Network

IP Multicast Traffic and Layer 2 Switches

The default behavior for a Layer 2 switch is to forward all multicast traffic to every port that belongs to the destination LAN on the switch. This behavior reduces the efficiency of the switch, whose purpose is to limit traffic to the ports that need to receive the data. This behavior requires a constraining mechanism to reduce unnecessary multicast traffic, which improves switch performance.

Cisco Group Management Protocol (CGMP), Router Group Management Protocol (RGMP), and IGMP snooping efficiently constrain IP multicast in a Layer 2 switching environment.

- CGMP and IGMP snooping are used on subnets that include end users or receiver clients.
- RGMP is used on routed segments that contain only routers, such as in a collapsed backbone.
- RGMP and CGMP cannot interoperate. However, Internet Group Management Protocol (IGMP) can interoperate with CGMP and RGMP snooping.

CGMP on Switches for IP Multicast

CGMP is a Cisco-developed protocol used on device connected to switches to perform tasks similar to those performed by IGMP. CGMP is necessary for those switches that do not distinguish between IP multicast data packets and IGMP report messages, both of which are addressed to the same group address at the MAC level. The switch can distinguish IGMP packets, but would need to use software on the switch, greatly impacting its performance.

You must configure CGMP on the multicast device and the Layer 2 switches. The result is that, with CGMP, IP multicast traffic is delivered only to those switch ports that are attached to interested receivers. All other ports that have not explicitly requested the traffic will not receive it unless these ports are connected to a multicast router. Multicast router ports must receive every IP multicast data packet.

Using CGMP, when a host joins a multicast group, it multicasts an unsolicited IGMP membership report message to the target group. The IGMP report is passed through the switch to the router for normal IGMP processing. The router (which must have CGMP enabled on this interface) receives the IGMP report and processes it as it normally would, but also creates a CGMP Join message and sends it to the switch. The Join message includes the MAC address of the end station and the MAC address of the group it has joined.

The switch receives this CGMP Join message and then adds the port to its content-addressable memory (CAM) table for that multicast group. All subsequent traffic directed to this multicast group is then forwarded out the port for that host.

The Layer 2 switches are designed so that several destination MAC addresses could be assigned to a single physical port. This design allows switches to be connected in a hierarchy and also allows many multicast destination addresses to be forwarded out a single port.

The device port also is added to the entry for the multicast group. Multicast device must listen to all multicast traffic for every group because IGMP control messages are also sent as multicast traffic. The rest of the multicast traffic is forwarded using the CAM table with the new entries created by CGMP.

IGMP Snooping

IGMP snooping is an IP multicast constraining mechanism that runs on a Layer 2 LAN switch. IGMP snooping requires the LAN switch to examine, or “snoop,” some Layer 3 information (IGMP Join/Leave messages) in the IGMP packets sent between the hosts and the router. When the switch receives the IGMP host report from a host for a particular multicast group, the switch adds the port number of the host to the associated multicast table entry. When the switch hears the IGMP Leave group message from a host, the switch removes the table entry of the host.

Because IGMP control messages are sent as multicast packets, they are indistinguishable from multicast data at Layer 2. A switch running IGMP snooping must examine every multicast data packet to determine if it contains any pertinent IGMP control information. IGMP snooping implemented on a low-end switch with a slow CPU could have a severe performance impact when data is sent at high rates. The solution is to implement IGMP snooping on high-end switches with special application-specific integrated circuits (ASICs) that can perform the IGMP checks in hardware. CGMP is a better option for low-end switches without special hardware.

Router-Port Group Management Protocol (RGMP)

CGMP and IGMP snooping are IP multicast constraining mechanisms designed to work on routed network segments that have active receivers. They both depend on IGMP control messages that are sent between the hosts and the routers to determine which switch ports are connected to interested receivers.

Switched Ethernet backbone network segments typically consist of several routers connected to a switch without any hosts on that segment. Because routers do not generate IGMP host reports, CGMP and IGMP snooping will not be able to constrain the multicast traffic, which will be flooded to every port on the VLAN. Routers instead generate Protocol Independent Multicast (PIM) messages to Join and Prune multicast traffic flows at a Layer 3 level.

Router-Port Group Management Protocol (RGMP) is an IP multicast constraining mechanism for router-only network segments. RGMP must be enabled on the routers and on the Layer 2 switches. A multicast router indicates that it is interested in receiving a data flow by sending an RGMP Join message for a particular group. The switch then adds the appropriate port to its forwarding table for that multicast group--similar to the way it handles a CGMP Join message. IP multicast data flows will be forwarded only to the interested router ports. When the router no longer is interested in that data flow, it sends an RGMP Leave message and the switch removes the forwarding entry.

If there are any routers that are not RGMP-enabled, they will continue to receive all multicast data.

How to Constrain Multicast in a Switched Ethernet Network

Configuring Switches for IP Multicast

If you have switching in your multicast network, consult the documentation for the switch you are working with for information about how to configure IP multicast.

Configuring IGMP Snooping

No configuration is required on the router. Consult the documentation for the switch you are working with to determine how to enable IGMP snooping and follow the provided instructions.

Enabling CGMP

CGMP is a protocol used on devices connected to switches to perform tasks similar to those performed by IGMP. CGMP is necessary because the switch cannot distinguish between IP multicast data packets and IGMP report messages, which are both at the MAC level and are addressed to the same group address.

Note

- CGMP should be enabled only on 802 or ATM media, or LAN emulation (LANE) over ATM.
- CGMP should be enabled only on devices connected to switches.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet1/1`	Selects an interface that is connected to hosts on which IGMPv3 can be enabled.
Step 4	**ip cgmp** [**proxy** | **router-only**] **Example:** `Device(config-if)# ip cgmp proxy`	Enables CGMP on an interface of a device connected to a switch. • The **proxy** keyword enables the CGMP proxy function. When enabled, any device that is not CGMP-capable will be advertised by the proxy router. The proxy router advertises the existence of other non-CGMP-capable devices by sending a CGMP Join message with the MAC address of the non-CGMP-capable device and group address of 0000.0000.0000.
Step 5	**end** **Example:** `Device(config-if)# end`	Ends the current configuration session and returns to EXEC mode.
Step 6	**clear ip cgmp** [*interface-type interface-number*] **Example:** `Device# clear ip cgmp`	(Optional) Clears all group entries from the caches of switches.

Configuring IP Multicast in a Layer 2 Switched Ethernet Network

Perform this task to configure IP multicast in a Layer 2 Switched Ethernet network using RGMP.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

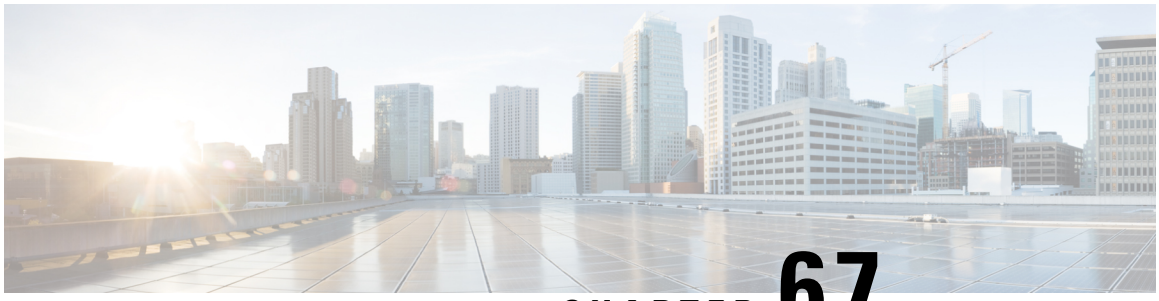
	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface Gigabitethernet1/1`	Selects an interface that is connected to hosts.
Step 4	**ip rgmp** **Example:** `Device(config-if)# ip rgmp`	Enables RGMP on Gigabit Ethernet interfaces.
Step 5	**end** **Example:** `Device(config-if)# end`	Ends the current configuration session and returns to EXEC mode.
Step 6	**debug ip rgmp** **Example:** `Device# debug ip rgmp`	(Optional) Logs debug messages sent by an RGMP-enabled device.
Step 7	**show ip igmp interface** **Example:** `Device# show ip igmp interface`	(Optional) Displays multicast-related information about an interface.

Configuration Examples for Constraining IP Multicast in a Switched Ethernet Network

Router Group Management Protocol Configuration Example

The following example shows how to configure RGMP on a router:

```
ip multicast-routing
ip pim sparse-mode
interface gigabitethernet 1/1
 ip rgmp
```



CHAPTER 67

Configuring Protocol Independent Multicast (PIM)

Prerequisites for PIM

Before you begin the PIM configuration process, decide which PIM mode to use. This is based on the applications you intend to support on your network. Use the following guidelines:

- In general, if the application is one-to-many or many-to-many in nature, then PIM-SM can be used successfully.
- For optimal one-to-many application performance, SSM is appropriate but requires IGMP version 3 support.

Before you configure PIM stub routing, check that you have met these conditions:

- You must have IP multicast routing configured on both the stub router and the central router. You must also have PIM mode configured on the uplink interface of the stub router.
- You must also configure either Enhanced Interior Gateway Routing Protocol (EIGRP) stub routing or Open Shortest Path First (OSPF) stub routing on the device.
- The PIM stub router does not route the transit traffic between the distribution routers. Unicast (EIGRP) stub routing enforces this behavior. You must configure unicast stub routing to assist the PIM stub router behavior.

Restrictions for PIM

The following are the restrictions for configuring PIM:

- Use ACLs to designate a specified port only as a multicast host port and not as a multicast router port. Multicast router control-packets received on this port are dropped.
- PIM nonbroadcast multiaccess (NBMA) mode is not supported on an ethernet interface.
- Hot Standby Router Protocol-aware (HSRP-aware) PIM is not supported.

PIMv1 and PIMv2 Interoperability

To avoid misconfiguring multicast routing on your device, review the information in this section.

The Cisco PIMv2 implementation provides interoperability and transition between Version 1 and Version 2, although there might be some minor problems.

You can upgrade to PIMv2 incrementally. PIM Versions 1 and 2 can be configured on different routers and multilayer switches within one network. Internally, all routers and multilayer switches on a shared media network must run the same PIM version. Therefore, if a PIMv2 device detects a PIMv1 device, the Version 2 device downgrades itself to Version 1 until all Version 1 devices have been shut down or upgraded.

PIMv2 uses the BSR to discover and announce RP-set information for each group prefix to all the routers and multilayer switches in a PIM domain. PIMv1, together with the Auto-RP feature, can perform the same tasks as the PIMv2 BSR. However, Auto-RP is a standalone protocol, separate from PIMv1, and is a proprietary Cisco protocol. PIMv2 is a standards track protocol in the IETF.

Note We recommend that you use PIMv2. The BSR function interoperates with Auto-RP on Cisco routers and multilayer switches.

When PIMv2 devices interoperate with PIMv1 devices, Auto-RP should have already been deployed. A PIMv2 BSR that is also an Auto-RP mapping agent automatically advertises the RP elected by Auto-RP. That is, Auto-RP sets its single RP on every router or multilayer switch in the group. Not all routers and switches in the domain use the PIMv2 hash function to select multiple RPs.

Sparse-mode groups in a mixed PIMv1 and PIMv2 region are possible because the Auto-RP feature in PIMv1 interoperates with the PIMv2 RP feature. Although all PIMv2 devices can also use PIMv1, we recommend that the RPs be upgraded to PIMv2. To ease the transition to PIMv2, we recommend:

- Using Auto-RP throughout the region.

If Auto-RP is not already configured in the PIMv1 regions, configure Auto-RP.

Restrictions for Configuring PIM Stub Routing

- Only directly connected multicast (IGMP) receivers and sources are allowed in the Layer 2 access domains. The PIM protocol is not supported in access domains.

- In a network using PIM stub routing, the only allowable route for IP traffic to the user is through a device that is configured with PIM stub routing.
- The redundant PIM stub router topology is not supported. Only the nonredundant access router topology is supported by the PIM stub feature.

Restrictions for Configuring Auto-RP and BSR

Take into consideration your network configuration, and the following restrictions when configuring Auto-RP and BSR:

Restrictions for Configuring Auto-RP

The following are restrictions for configuring Auto-RP (if used in your network configuration):

- If routed interfaces are configured in sparse mode, Auto-RP can still be used if all devices are configured with a manual RP address for the Auto-RP groups.
- If routed interfaces are configured in sparse mode and you enter the **ip pim autorp listener** global configuration command, Auto-RP can still be used even if all devices are not configured with a manual RP address for the Auto-RP groups.

Restrictions for Configuring BSR

The following are the restrictions for configuring BSR (if used in your network configuration):

- Configure the candidate BSRs as the RP-mapping agents for Auto-RP.
- For group prefixes advertised through Auto-RP, the PIMv2 BSR mechanism should not advertise a subrange of these group prefixes served by a different set of RPs. In a mixed PIMv1 and PIMv2 domain, have backup RPs serve the same group prefixes. This prevents the PIMv2 DRs from selecting a different RP from those PIMv1 DRs, due to the longest match lookup in the RP-mapping database.

Restrictions and Guidelines for Configuring Auto-RP and BSR

The following are restrictions for configuring Auto-RP and BSR (if used in your network configuration):

- If your network is all Cisco routers and multilayer switches, you can use either Auto-RP or BSR.
- If you have non-Cisco routers in your network, you must use BSR.
- If you have Cisco PIMv1 and PIMv2 routers and multilayer switches and non-Cisco routers, you must use both Auto-RP and BSR. If your network includes routers from other vendors, configure the Auto-RP mapping agent and the BSR on a Cisco PIMv2 device. Ensure that no PIMv1 device is located in the path a between the BSR and a non-Cisco PIMv2 device.

Note There are two approaches to using PIMv2. You can use Version 2 exclusively in your network or migrate to Version 2 by employing a mixed PIM version environment.

- Because bootstrap messages are sent hop-by-hop, a PIMv1 device prevents these messages from reaching all routers and multilayer switches in your network. Therefore, if your network has a PIMv1 device in it and only Cisco routers and multilayer switches, it is best to use Auto-RP.
- If you have a network that includes non-Cisco routers, configure the Auto-RP mapping agent and the BSR on a Cisco PIMv2 router or multilayer switch. Ensure that no PIMv1 device is on the path between the BSR and a non-Cisco PIMv2 router.
- If you have non-Cisco PIMv2 routers that need to interoperate with Cisco PIMv1 routers and multilayer switches, both Auto-RP and a BSR are required. We recommend that a Cisco PIMv2 device be both the Auto-RP mapping agent and the BSR.

Restrictions for Auto-RP Enhancement

The simultaneous deployment of Auto-RP and bootstrap router (BSR) is not supported.

Information about PIM

Protocol Independent Multicast Overview

The Protocol Independent Multicast (PIM) protocol maintains the current IP multicast service mode of receiver-initiated membership. PIM is not dependent on a specific unicast routing protocol; it is IP routing protocol independent and can leverage whichever unicast routing protocols are used to populate the unicast routing table, including Enhanced Interior Gateway Routing Protocol (EIGRP), Open Shortest Path First (OSPF), and static routes. PIM uses unicast routing information to perform the multicast forwarding function.

Although PIM is called a multicast routing protocol, it actually uses the unicast routing table to perform the reverse path forwarding (RPF) check function instead of building up a completely independent multicast routing table. Unlike other routing protocols, PIM does not send and receive routing updates between routers.

PIM is defined in RFC 4601, Protocol Independent Multicast - Sparse Mode (PIM-SM)

PIM Versions

PIMv2 includes these improvements over PIMv1:

- A single, active rendezvous point (RP) exists per multicast group, with multiple backup RPs. This single RP compares to multiple active RPs for the same group in PIMv1.
- A bootstrap router (BSR) provides a fault-tolerant, automated RP discovery and distribution function that enables routers and multilayer switches to dynamically learn the group-to-RP mappings.
- PIM join and prune messages have more flexible encoding for multiple address families.
- A more flexible hello packet format replaces the query packet to encode current and future capability options.
- Register messages sent to an RP specify whether they are sent by a border router or a designated router.
- PIM packets are no longer inside IGMP packets; they are standalone packets.

Multicast Source Discovery Protocol (MSDP)

Multicast Source Discovery Protocol (MSDP) is used for inter-domain source discovery when PIM SM is used. Each PIM administrative domain has its own RP. In order for the RP in one domain to signal new sources to the RP in the other domain, MSDP is used.

When RP in a domain receives a PIM register message for a new source, with MSDP configured it sends a new source-active (SA) message to all its MSDP peers in other domains. Each intermediate MSDP peer floods this SA message away from the originating RP. The MSDP peers install this SA message in their MSDP sa-cache. If the RPs in other domains have any join requests for the group in the SA message (indicated by the presence of a (*,G) entry with non empty outgoing interface list), the domain is interested in the group, and the RP triggers an (S,G) join toward the source.

PIM Sparse Mode

PIM sparse mode (PIM-SM) uses a pull model to deliver multicast traffic. Only network segments with active receivers that have explicitly requested the data will receive the traffic.

Sparse mode interfaces are added to the multicast routing table only when periodic Join messages are received from downstream routers, or when a directly connected member is on the interface. When forwarding from a LAN, sparse mode operation occurs if an RP is known for the group. If so, the packets are encapsulated and sent toward the RP. If the multicast traffic from a specific source is sufficient, the first hop router of the receiver may send Join messages toward the source to build a source-based distribution tree.

PIM-SM distributes information about active sources by forwarding data packets on the shared tree. Because PIM-SM uses shared trees (at least, initially), it requires the use of a rendezvous point (RP). The RP must be administratively configured in the network. See the Rendezvous Points, on page 911 section for more information.

In sparse mode, a router assumes that other routers do not want to forward multicast packets for a group, unless there is an explicit request for the traffic. When hosts join a multicast group, the directly connected routers send PIM Join messages toward the RP. The RP keeps track of multicast groups. Hosts that send multicast packets are registered with the RP by the first hop router of that host. The RP then sends Join messages toward the source. At this point, packets are forwarded on a shared distribution tree. If the multicast traffic from a specific source is sufficient, the first hop router of the host may send Join messages toward the source to build a source-based distribution tree.

Sources register with the RP and then data is forwarded down the shared tree to the receivers. The edge routers learn about a particular source when they receive data packets on the shared tree from that source through the RP. The edge router then sends PIM (S,G) Join messages toward that source. Each router along the reverse path compares the unicast routing metric of the RP address to the metric of the source address. If the metric for the source address is better, it will forward a PIM (S,G) Join message toward the source. If the metric for the RP is the same or better, then the PIM (S,G) Join message will be sent in the same direction as the RP. In this case, the shared tree and the source tree would be considered congruent.

If the shared tree is not an optimal path between the source and the receiver, the routers dynamically create a source tree and stop traffic from flowing down the shared tree. This behavior is the default behavior in software. Network administrators can force traffic to stay on the shared tree by using the **ip pim spt-threshold infinity** command.

PIM-SM scales well to a network of any size, including those with WAN links. The explicit join mechanism prevents unwanted traffic from flooding the WAN links.

PIM Stub Routing

The PIM stub routing feature, available in all of the device software images, reduces resource usage by moving routed traffic closer to the end user.

The PIM stub routing feature supports multicast routing between the distribution layer and the access layer. It supports two types of PIM interfaces, uplink PIM interfaces, and PIM passive interfaces. A routed interface configured with the PIM passive mode does not pass or forward PIM control traffic, it only passes and forwards IGMP traffic.

In a network using PIM stub routing, the only allowable route for IP traffic to the user is through a device that is configured with PIM stub routing. PIM passive interfaces are connected to Layer 2 access domains, such as VLANs, or to interfaces that are connected to other Layer 2 devices. Only directly connected multicast (IGMP) receivers and sources are allowed in the Layer 2 access domains. The PIM passive interfaces do not send or process any received PIM control packets.

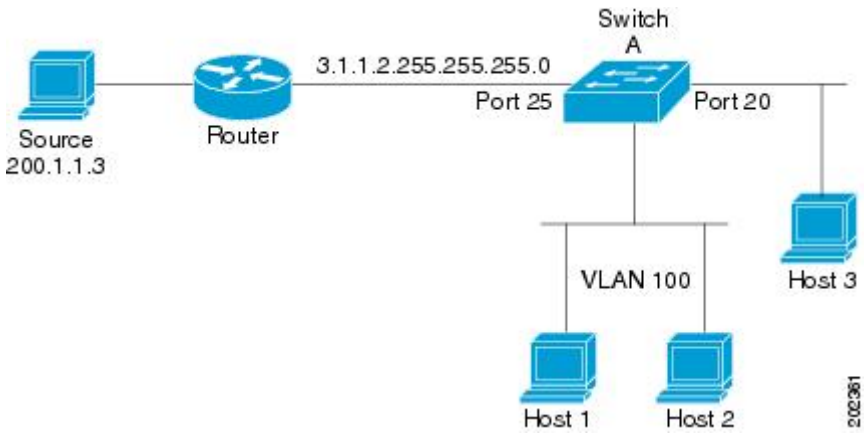
When using PIM stub routing, you should configure the distribution and remote routers to use IP multicast routing and configure only the device as a PIM stub router. The device does not route transit traffic between distribution routers. You also need to configure a routed uplink port on the device. The device uplink port cannot be used with SVIs. If you need PIM for an SVI uplink port, you should upgrade to the Network Advantage license.

Note You must also configure EIGRP stub routing when configuring PIM stub routing on the device

The redundant PIM stub router topology is not supported. The redundant topology exists when there is more than one PIM router forwarding multicast traffic to a single access domain. PIM messages are blocked, and the PIM asset and designated router election mechanisms are not supported on the PIM passive interfaces. Only the nonredundant access router topology is supported by the PIM stub feature. By using a nonredundant topology, the PIM passive interface assumes that it is the only interface and designated router on that access domain.

Figure 72: PIM Stub Router Configuration

In the following figure, the Device A routed uplink port 25 is connected to the router and PIM stub routing is enabled on the VLAN 100 interfaces and on Host 3. This configuration allows the directly connected hosts to receive traffic from multicast source 200.1.1.3.



Rendezvous Points

A rendezvous point (RP) is a role that a device performs when operating in Protocol Independent Multicast (PIM) Sparse Mode (SM). An RP is required only in networks running PIM SM. In the PIM-SM model, only network segments with active receivers that have explicitly requested multicast data will be forwarded the traffic.

An RP acts as the meeting place for sources and receivers of multicast data. In a PIM-SM network, sources must send their traffic to the RP. This traffic is then forwarded to receivers down a shared distribution tree. By default, when the first hop device of the receiver learns about the source, it will send a Join message directly to the source, creating a source-based distribution tree from the source to the receiver. This source tree does not include the RP unless the RP is located within the shortest path between the source and receiver.

In most cases, the placement of the RP in the network is not a complex decision. By default, the RP is needed only to start new sessions with sources and receivers. Consequently, the RP experiences little overhead from traffic flow or processing. In PIM version 2, the RP performs less processing than in PIM version 1 because sources must only periodically register with the RP to create state.

Auto-RP

In the first version of PIM-SM, all leaf routers (routers directly connected to sources or receivers) were required to be manually configured with the IP address of the RP. This type of configuration is also known as static RP configuration. Configuring static RPs is relatively easy in a small network, but it can be laborious in a large, complex network.

Following the introduction of PIM-SM version 1, Cisco implemented a version of PIM-SM with the Auto-RP feature. Auto-RP automates the distribution of group-to-RP mappings in a PIM network. Auto-RP has the following benefits:

- Configuring the use of multiple RPs within a network to serve different groups is easy.
- Auto-RP allows load splitting among different RPs and arrangement of RPs according to the location of group participants.
- Auto-RP avoids inconsistent, manual RP configurations that can cause connectivity problems.

Multiple RPs can be used to serve different group ranges or serve as backups to each other. For Auto-RP to work, a router must be designated as an RP-mapping agent, which receives the RP-announcement messages from the RPs and arbitrates conflicts. The RP-mapping agent then sends the consistent group-to-RP mappings to all other routers. Thus, all routers automatically discover which RP to use for the groups they support.

Note If router interfaces are configured in sparse mode, Auto-RP can still be used if all routers are configured with a static RP address for the Auto-RP groups.

To make Auto-RP work, a router must be designated as an RP mapping agent, which receives the RP announcement messages from the RPs and arbitrates conflicts. Thus, all routers automatically discover which RP to use for the groups they support. The Internet Assigned Numbers Authority (IANA) has assigned two group addresses, 224.0.1.39 and 224.0.1.40, for Auto-RP. One advantage of Auto-RP is that any change to the RP designation must be configured only on the routers that are RPs and not on the leaf routers. Another advantage of Auto-RP is that it offers the ability to scope the RP address within a domain. Scoping can be achieved by defining the time-to-live (TTL) value allowed for the Auto-RP advertisements.

Each method for configuring an RP has its own strengths, weaknesses, and level of complexity. In conventional IP multicast network scenarios, we recommend using Auto-RP to configure RPs because it is easy to configure, well-tested, and stable. The alternative ways to configure an RP are static RP, Auto-RP, and bootstrap router.

The Role of Auto-RP in a PIM Network

Auto-RP automates the distribution of group-to-rendezvous point (RP) mappings in a PIM network. To make Auto-RP work, a device must be designated as an RP mapping agent, which receives the RP announcement messages from the RPs and arbitrates conflicts.

Thus, all routers automatically discover which RP to use for the groups they support. The Internet Assigned Numbers Authority (IANA) has assigned two group addresses, 224.0.1.39 and 224.0.1.40, for Auto-RP.

The mapping agent receives announcements of intention to become the RP from Candidate-RPs. The mapping agent then announces the winner of the RP election. This announcement is made independently of the decisions by the other mapping agents.

Multicast Boundaries

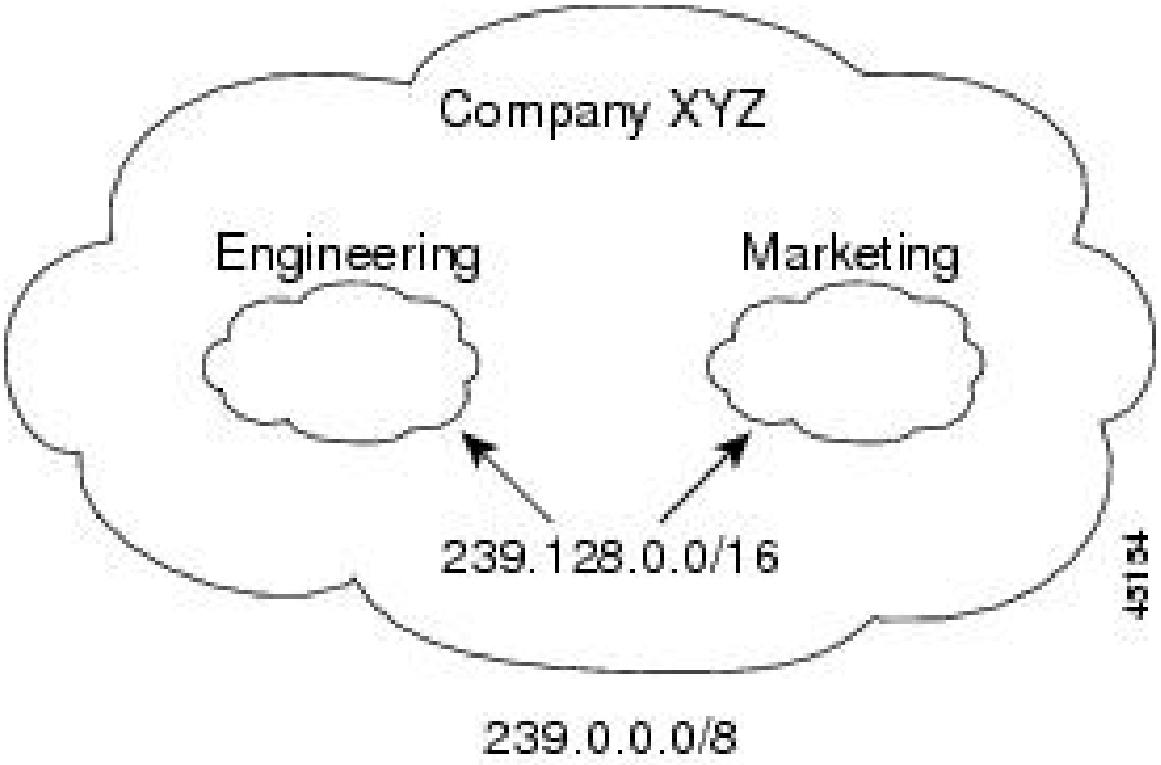
Administratively-scoped boundaries can be used to limit the forwarding of multicast traffic outside of a domain or subdomain. This approach uses a special range of multicast addresses, called administratively-scoped addresses, as the boundary mechanism. If you configure an administratively-scoped boundary on a routed interface, multicast traffic whose multicast group addresses fall in this range cannot enter or exit this interface, which provides a firewall for multicast traffic in this address range.

Note Multicast boundaries and TTL thresholds control the scoping of multicast domains; however, TTL thresholds are not supported by the device. You should use multicast boundaries instead of TTL thresholds to limit the forwarding of multicast traffic outside of a domain or a subdomain.

Figure 73: Administratively-Scoped Boundaries

The following figure shows that Company XYZ has an administratively-scoped boundary set for the multicast address range 239.0.0.0/8 on all routed interfaces at the perimeter of its network. This boundary prevents any multicast traffic in the range 239.0.0.0 through 239.255.255.255 from entering or leaving the network. Similarly, the engineering and marketing departments have an administratively-scoped boundary of 239.128.0.0/16 around the perimeter of their networks. This boundary prevents multicast traffic in the range of 239.128.0.0

through 239.128.255.255 from entering or leaving their respective networks.



You can define an administratively-scoped boundary on a routed interface for multicast group addresses. A standard access list defines the range of addresses affected. When a boundary is defined, no multicast data packets are allowed to flow across the boundary from either direction. The boundary allows the same multicast group address to be reused in different administrative domains.

The IANA has designated the multicast address range 239.0.0.0 to 239.255.255.255 as the administratively-scoped addresses. This range of addresses can then be reused in domains administered by different organizations. The addresses would be considered local, not globally unique.

You can configure the **filter-autorp** keyword to examine and filter Auto-RP discovery and announcement messages at the administratively scoped boundary. Any Auto-RP group range announcements from the Auto-RP packets that are denied by the boundary access control list (ACL) are removed. An Auto-RP group range announcement is permitted and passed by the boundary only if all addresses in the Auto-RP group range are permitted by the boundary ACL. If any address is not permitted, the entire group range is filtered and removed from the Auto-RP message before the Auto-RP message is forwarded.

Sparse-Dense Mode for Auto-RP

A prerequisite of Auto-RP is that all interfaces must be configured in sparse-dense mode using the **ip pim sparse-dense-mode** interface configuration command. An interface configured in sparse-dense mode is treated in either sparse mode or dense mode of operation, depending on which mode the multicast group operates. If a multicast group has a known RP, the interface is treated in sparse mode. If a group has no known RP, by default the interface is treated in dense mode and data will be flooded over this interface. (You can prevent dense-mode fallback; see the module “Configuring Basic IP Multicast.”)

To successfully implement Auto-RP and prevent any groups other than 224.0.1.39 and 224.0.1.40 from operating in dense mode, we recommend configuring a “sink RP” (also known as “RP of last resort”). A sink

RP is a statically configured RP that may or may not actually exist in the network. Configuring a sink RP does not interfere with Auto-RP operation because, by default, Auto-RP messages supersede static RP configurations. We recommend configuring a sink RP for all possible multicast groups in your network, because it is possible for an unknown or unexpected source to become active. If no RP is configured to limit source registration, the group may revert to dense mode operation and be flooded with data.

Auto RP Benefits

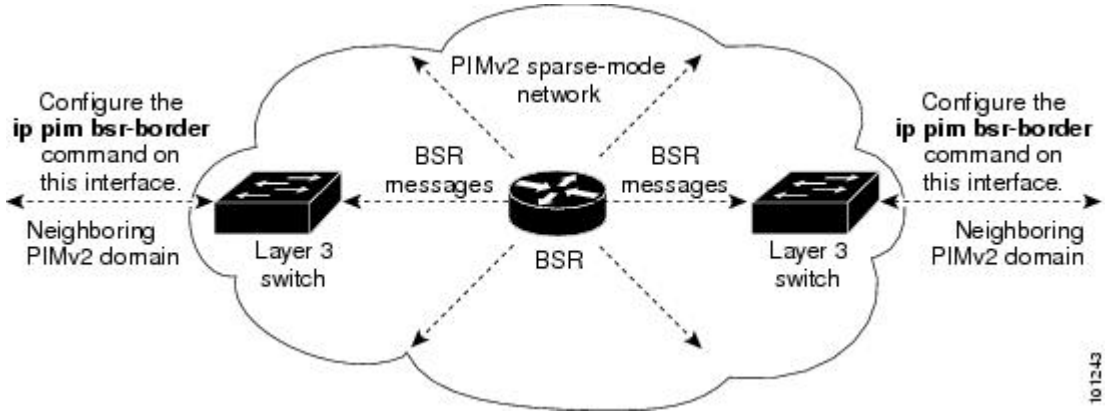
Benefits of Auto-RP in a PIM Network

- Auto-RP allows any change to the RP designation to be configured only on the devices that are RPs, not on the leaf routers.
- Auto-RP offers the ability to scope the RP address within a domain.

PIM Domain Border

As IP multicast becomes more widespread, the chance of one PIMv2 domain bordering another PIMv2 domain increases. Because two domains probably do not share the same set of RPs, BSR, candidate RPs, and candidate BSRs, you need to constrain PIMv2 BSR messages from flowing into or out of the domain. Allowing messages to leak across the domain borders could adversely affect the normal BSR election mechanism and elect a single BSR across all bordering domains and comingle candidate RP advertisements, resulting in the election of RPs in the wrong domain.

This figure displays how you can configure the PIM domain border by using the **ip pim bsr-border** command.



PIMv2 Bootstrap Router

PIMv2 Bootstrap Router (BSR) is another method to distribute group-to-RP mapping information to all PIM routers and multilayer devices in the network. It eliminates the need to manually configure RP information in every router and switch in the network. However, instead of using IP multicast to distribute group-to-RP mapping information, BSR uses hop-by-hop flooding of special BSR messages to distribute the mapping information.

The BSR is elected from a set of candidate routers and switches in the domain that have been configured to function as BSRs. The election mechanism is similar to the root-bridge election mechanism used in bridged LANs. The BSR election is based on the BSR priority of the device contained in the BSR messages that are sent hop-by-hop through the network. Each BSR device examines the message and forwards out all interfaces only the message that has either a higher BSR priority than its BSR priority or the same BSR priority, but with a higher BSR IP address. Using this method, the BSR is elected.

The elected BSR sends BSR messages with a TTL of 1. Neighboring PIMv2 routers or multilayer devices receive the BSR message and multicast it out all other interfaces (except the one on which it was received) with a TTL of 1. In this way, BSR messages travel hop-by-hop throughout the PIM domain. Because BSR messages contain the IP address of the current BSR, the flooding mechanism enables candidate RPs to automatically learn which device is the elected BSR.

Candidate RPs send candidate RP advertisements showing the group range for which they are responsible to the BSR, which stores this information in its local candidate-RP cache. The BSR periodically advertises the contents of this cache in BSR messages to all other PIM devices in the domain. These messages travel hop-by-hop through the network to all routers and switches, which store the RP information in the BSR message in their local RP cache. The routers and switches select the same RP for a given group because they all use a common RP hashing algorithm.

Multicast Forwarding

Forwarding of multicast traffic is accomplished by multicast-capable routers. These routers create distribution trees that control the path that IP multicast traffic takes through the network in order to deliver traffic to all receivers.

Multicast traffic flows from the source to the multicast group over a distribution tree that connects all of the sources to all of the receivers in the group. This tree may be shared by all sources (a shared tree) or a separate distribution tree can be built for each source (a source tree). The shared tree may be one-way or bidirectional.

Before describing the structure of source and shared trees, it is helpful to explain the notations that are used in multicast routing tables. These notations include the following:

- (S,G) = (unicast source for the multicast group G, multicast group G)
- (*,G) = (any source for the multicast group G, multicast group G)

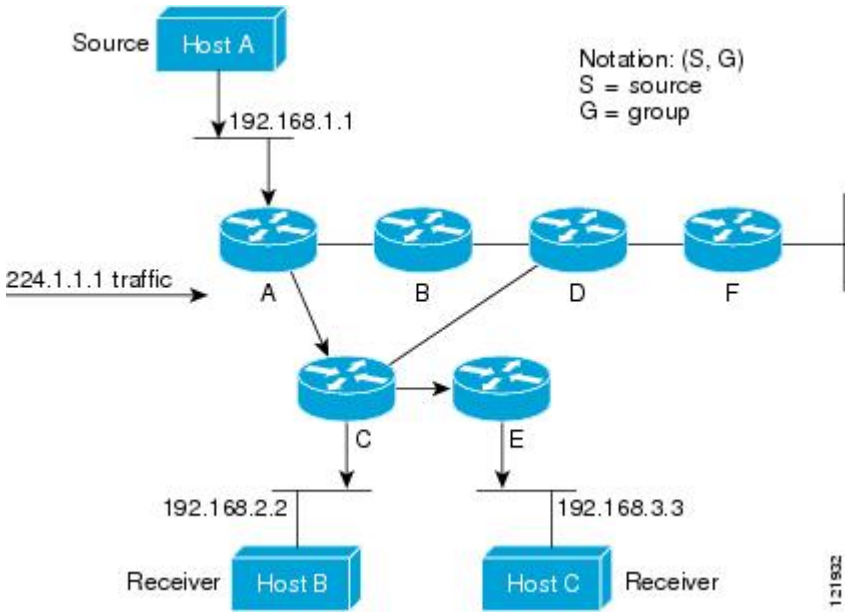
The notation of (S,G), pronounced “S comma G,” enumerates a shortest path tree where S is the IP address of the source and G is the multicast group address.

Shared trees are (*,G) and the source trees are (S,G) and always routed at the sources.

Multicast Distribution Source Tree

The simplest form of a multicast distribution tree is a source tree. A source tree has its root at the source host and has branches forming a spanning tree through the network to the receivers. Because this tree uses the shortest path through the network, it is also referred to as a shortest path tree (SPT).

The figure shows an example of an SPT for group 224.1.1.1 rooted at the source, Host A, and connecting two receivers, Hosts B and C.



Using standard notation, the SPT for the example shown in the figure would be (192.168.1.1, 224.1.1.1).

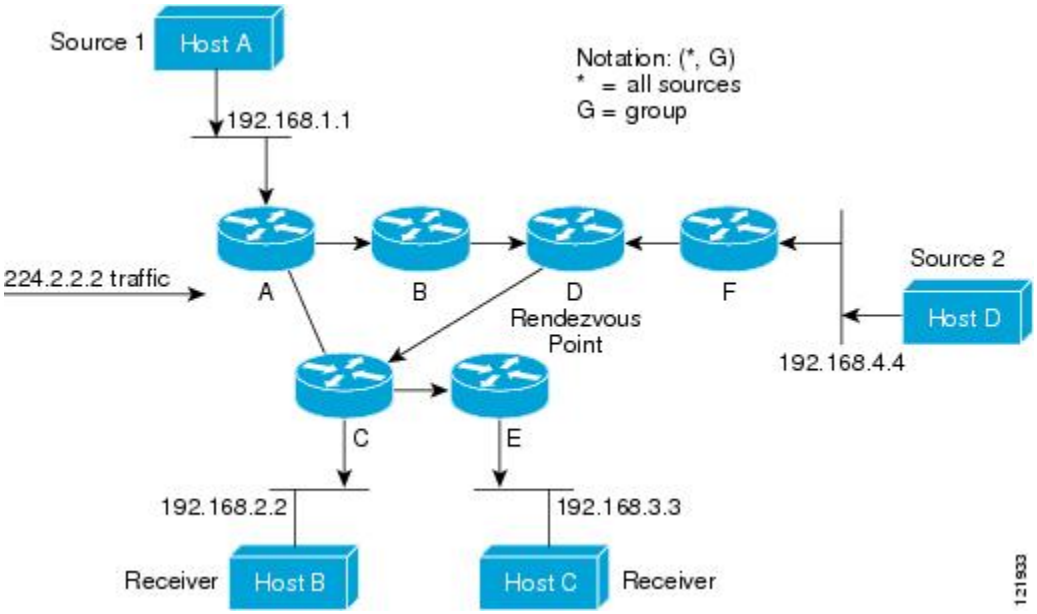
The (S,G) notation implies that a separate SPT exists for each individual source sending to each group--which is correct.

Multicast Distribution Shared Tree

Unlike source trees that have their root at the source, shared trees use a single common root placed at some chosen point in the network. This shared root is called a rendezvous point (RP).

The following figure shows a shared tree for the group 224.2.2.2 with the root located at Router D. This shared tree is unidirectional. Source traffic is sent towards the RP on a source tree. The traffic is then forwarded down the shared tree from the RP to reach all of the receivers (unless the receiver is located between the source and the RP, in which case it will be serviced directly).

Figure 74: Shared Tree



In this example, multicast traffic from the sources, Hosts A and D, travels to the root (Router D) and then down the shared tree to the two receivers, Hosts B and C. Because all sources in the multicast group use a common shared tree, a wildcard notation written as (*, G), pronounced "star comma G", represents the tree. In this case, * means all sources, and G represents the multicast group. Therefore, the shared tree shown in the figure would be written as (*, 224.2.2.2).

Both source trees and shared trees are loop-free. Messages are replicated only where the tree branches. Members of multicast groups can join or leave at any time; therefore the distribution trees must be dynamically updated. When all the active receivers on a particular branch stop requesting the traffic for a particular multicast group, the routers prune that branch from the distribution tree and stop forwarding traffic down that branch. If one receiver on that branch becomes active and requests the multicast traffic, the router will dynamically modify the distribution tree and start forwarding traffic again.

Source Tree Advantage

Source trees have the advantage of creating the optimal path between the source and the receivers. This advantage guarantees the minimum amount of network latency for forwarding multicast traffic. However, this optimization comes at a cost. The routers must maintain path information for each source. In a network that has thousands of sources and thousands of groups, this overhead can quickly become a resource issue on the routers. Memory consumption from the size of the multicast routing table is a factor that network designers must take into consideration.

Shared Tree Advantage

Shared trees have the advantage of requiring the minimum amount of state in each router. This advantage lowers the overall memory requirements for a network that only allows shared trees. The disadvantage of shared trees is that under certain circumstances the paths between the source and receivers might not be the optimal paths, which might introduce some latency in packet delivery. For example, in the figure above the shortest path between Host A (source 1) and Host B (a receiver) would be Router A and Router C. Because we are using Router D as the root for a shared tree, the traffic must traverse Routers A, B, D and then C.

Network designers must carefully consider the placement of the rendezvous point (RP) when implementing a shared tree-only environment.

In unicast routing, traffic is routed through the network along a single path from the source to the destination host. A unicast router does not consider the source address; it considers only the destination address and how to forward the traffic toward that destination. The router scans through its routing table for the destination address and then forwards a single copy of the unicast packet out the correct interface in the direction of the destination.

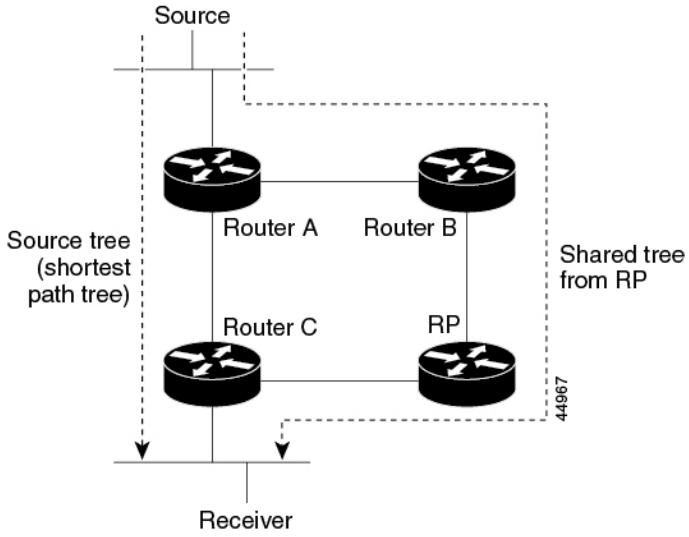
In multicast forwarding, the source is sending traffic to an arbitrary group of hosts that are represented by a multicast group address. The multicast router must determine which direction is the upstream direction (toward the source) and which one is the downstream direction (or directions) toward the receivers. If there are multiple downstream paths, the router replicates the packet and forwards it down the appropriate downstream paths (best unicast route metric)--which is not necessarily all paths. Forwarding multicast traffic away from the source, rather than to the receiver, is called Reverse Path Forwarding (RPF). RPF is described in the following section.

PIM Shared Tree and Source Tree

By default, members of a group receive data from senders to the group across a single data-distribution tree rooted at the RP.

Figure 75: Shared Tree and Source Tree (Shortest-Path Tree)

The following figure shows this type of shared-distribution tree. Data from senders is delivered to the RP for distribution to group members joined to the shared tree.



If the data rate warrants, leaf routers (routers without any downstream connections) on the shared tree can use the data distribution tree rooted at the source. This type of distribution tree is called a shortest-path tree or source tree. By default, the software device to a source tree upon receiving the first data packet from a source.

This process describes the move from a shared tree to a source tree:

1. A receiver joins a group; leaf Router C sends a join message toward the RP.
2. The RP puts a link to Router C in its outgoing interface list.
3. A source sends data; Router A encapsulates the data in a register message and sends it to the RP.

4. The RP forwards the data down the shared tree to Router C and sends a join message toward the source. At this point, data might arrive twice at Router C, once encapsulated and once natively.
5. When data arrives natively (unencapsulated) at the RP, it sends a register-stop message to Router A.
6. By default, reception of the first data packet prompts Router C to send a join message toward the source.
7. When Router C receives data on (S, G), it sends a prune message for the source up the shared tree.
8. The RP deletes the link to Router C from the outgoing interface of (S, G). The RP triggers a prune message toward the source.

Join and prune messages are sent for sources and RPs. They are sent hop-by-hop and are processed by each PIM device along the path to the source or RP. Register and register-stop messages are not sent hop-by-hop. They are sent by the designated router that is directly connected to a source and are received by the RP for the group.

Multiple sources sending to groups use the shared tree. You can configure the PIM device to stay on the shared tree.

The change from shared to source tree happens when the first data packet arrives at the last-hop router. This change depends upon the threshold that is configured by using the **ip pim spt-threshold** global configuration command.

The shortest-path tree requires more memory than the shared tree but reduces delay. You may want to postpone its use. Instead of allowing the leaf router to immediately move to the shortest-path tree, you can specify that the traffic must first reach a threshold.

You can configure when a PIM leaf router should join the shortest-path tree for a specified group. If a source sends at a rate greater than or equal to the specified kbps rate, the multilayer switch triggers a PIM join message toward the source to construct a source tree (shortest-path tree). If the traffic rate from the source drops below the threshold value, the leaf router switches back to the shared tree and sends a prune message toward the source.

You can specify to which groups the shortest-path tree threshold applies by using a group list (a standard access list). If a value of 0 is specified or if the group list is not used, the threshold applies to all groups.

Reverse Path Forwarding

In unicast routing, traffic is routed through the network along a single path from the source to the destination host. A unicast router does not consider the source address; it considers only the destination address and how to forward the traffic toward that destination. The router scans through its routing table for the destination network and then forwards a single copy of the unicast packet out the correct interface in the direction of the destination.

In multicast forwarding, the source is sending traffic to an arbitrary group of hosts that are represented by a multicast group address. The multicast router must determine which direction is the upstream direction (toward the source) and which one is the downstream direction (or directions) toward the receivers. If there are multiple downstream paths, the router replicates the packet and forwards it down the appropriate downstream paths (best unicast route metric)--which is not necessarily all paths. Forwarding multicast traffic away from the source, rather than to the receiver, is called Reverse Path Forwarding (RPF). RPF is an algorithm used for forwarding multicast datagrams.

Protocol Independent Multicast (PIM) uses the unicast routing information to create a distribution tree along the reverse path from the receivers towards the source. The multicast routers then forward packets along the distribution tree from the source to the receivers. RPF is a key concept in multicast forwarding. It enables

routers to correctly forward multicast traffic down the distribution tree. RPF makes use of the existing unicast routing table to determine the upstream and downstream neighbors. A router will forward a multicast packet only if it is received on the upstream interface. This RPF check helps to guarantee that the distribution tree will be loop-free.

RPF Check

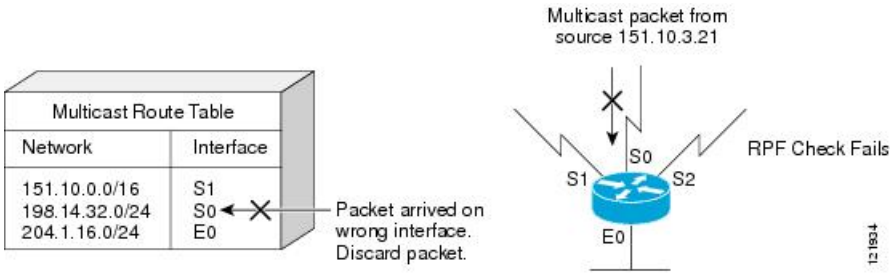
When a multicast packet arrives at a router, the router performs an RPF check on the packet. If the RPF check succeeds, the packet is forwarded. Otherwise, it is dropped.

For traffic flowing down a source tree, the RPF check procedure works as follows:

1. The router looks up the source address in the unicast routing table to determine if the packet has arrived on the interface that is on the reverse path back to the source.
2. If the packet has arrived on the interface leading back to the source, the RPF check succeeds and the packet is forwarded out the interfaces present in the outgoing interface list of a multicast routing table entry.
3. If the RPF check in Step 2 fails, the packet is dropped.

The figure shows an example of an unsuccessful RPF check.

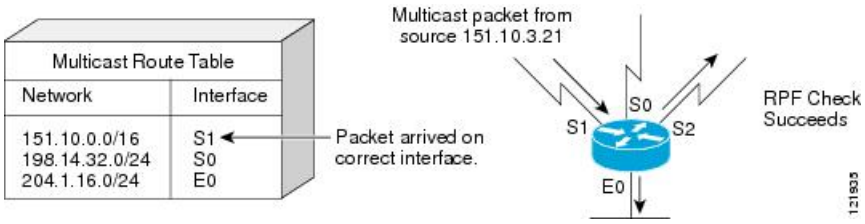
Figure 76: RPF Check Fails



As the figure illustrates, a multicast packet from source 151.10.3.21 is received on serial interface 0 (S0). A check of the unicast route table shows that S1 is the interface this router would use to forward unicast data to 151.10.3.21. Because the packet has arrived on interface S0, the packet is discarded.

The figure shows an example of a successful RPF check.

Figure 77: RPF Check Succeeds



In this example, the multicast packet has arrived on interface S1. The router refers to the unicast routing table and finds that S1 is the correct interface. The RPF check passes, and the packet is forwarded.

PIM uses both source trees and RP-rooted shared trees to forward datagrams. The RPF check is performed differently for each:

- If a PIM router or multilayer switch has a source-tree state (that is, an (S, G) entry is present in the multicast routing table), it performs the RPF check against the IP address of the source of the multicast packet.
- If a PIM router or multilayer switch has a shared-tree state (and no explicit source-tree state), it performs the RPF check on the RP address (which is known when members join the group).

Note DVMRP is not supported on the switch.

Sparse-mode PIM uses the RPF lookup function to decide where it needs to send joins and prunes:

- (S, G) joins (which are source-tree states) are sent toward the source.
- (*,G) joins (which are shared-tree states) are sent toward the RP.

Default PIM Routing Configuration

This table displays the default PIM routing configuration for the device.

Table 75: Default Multicast Routing Configuration

Feature	Default Setting
Multicast routing	Disabled on all interfaces.
PIM version	Version 2.
PIM mode	No mode is defined.
PIM stub routing	None configured.
PIM RP address	None configured.
PIM domain border	Disabled.
PIM multicast boundary	None.
Candidate BSRs	Disabled.
Candidate RPs	Disabled.
Shortest-path tree threshold rate	0 kb/s.
PIM router query message interval	30 seconds.

How to Configure PIM

Enabling PIM Stub Routing

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface on which you want to enable PIM stub routing, and enters interface configuration mode. • A routed port—A physical port that has been configured as a Layer 3 port by entering the **no switchport** interface configuration command. • An SVI—A VLAN interface created by using the **interface vlan** *vlan-id* global configuration command.
Step 4	**ip pim passive** **Example:** `Device(config-if)# ip pim passive`	Configures the PIM stub feature on the interface.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show ip pim interface** **Example:** `Device# show ip pim interface`	(Optional) Displays the PIM stub that is enabled on each interface.
Step 7	**show ip igmp groups detail** **Example:** `Device# show ip igmp groups detail`	(Optional) Displays the interested clients that have joined the specific multicast source group.
Step 8	**show ip mroute** **Example:** `Device# show ip mroute`	(Optional) Displays the IP multicast routing table.
Step 9	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 10	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Rendezvous Point

You must have a rendezvous point (RP), if the interface is in sparse-dense mode and if you want to handle the group as a sparse group. You can use these methods:

- By manually assigning an RP to multicast groups.
- As a standalone, Cisco-proprietary protocol separate from PIMv1, which includes:
- By using a standards track protocol in the Internet Engineering Task Force (IETF), which includes configuring PIMv2 BSR .

Note You can use Auto-RP, BSR, or a combination of both, depending on the PIM version that you are running and the types of routers in your network. For information about working with different PIM versions in your network, see PIMv1 and PIMv2 Interoperability, on page 906.

Manually Assigning an RP to Multicast Groups

If the rendezvous point (RP) for a group is learned through a dynamic mechanism (such as Auto-RP or BSR), you need not perform this task for that RP.

Senders of multicast traffic announce their existence through register messages received from the source first-hop router (designated router) and forwarded to the RP. Receivers of multicast packets use RPs to join a multicast group by using explicit join messages.

Note RPs are not members of the multicast group; they serve as a *meeting place* for multicast sources and group members.

You can configure a single RP for multiple groups defined by an access list. If there is no RP configured for a group, the multilayer switch responds to the group as dense and uses the dense-mode PIM techniques.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip pim rp-address** *ip-address* [*access-list-number*] [**override**] **Example:** `Device(config)# ip pim rp-address 10.1.1.1 20 override`	Configures the address of a PIM RP. By default, no PIM RP address is configured. You must configure the IP address of RPs on all routers and multilayer switches (including the RP). **Note** If there is no RP configured for a group, the device treats the group as dense, using the dense-mode PIM techniques. A PIM device can be an RP for more than one group. Only one RP address can be used at a time within a PIM domain. The access list conditions specify for which groups the device is an RP.

	Command or Action	Purpose
		• For *ip-address*, enter the unicast address of the RP in dotted-decimal notation. • (Optional) For *access-list-number*, enter an IP standard access list number from 1 to 99. If no access list is configured, the RP is used for all groups. • (Optional) The **override** keyword indicates that if there is a conflict between the RP configured with this command and one learned by Auto-RP or BSR, the RP configured with this command prevails.
Step 4	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] **Example:** `Device(config)# access-list 25 permit 10.5.0.1 255.224.0.0`	Creates a standard access list, repeating the command as many times as necessary. • For *access-list-number*, enter the access list number specified in Step 2. • The **deny** keyword denies access if the conditions are matched. • The **permit** keyword permits access if the conditions are matched. • For *source*, enter the multicast group address for which the RP should be used. • (Optional) For *source-wildcard*, enter the wildcard bits in dotted decimal notation to be applied to the source. Place ones in the bit positions that you want to ignore. The access list is always terminated by an implicit deny statement for everything.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:**	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`Device# copy running-config startup-config`	

Setting Up Auto-RP in a New Internetwork

Note Omit Step 3 in the following procedure, if you want to configure a PIM router as the RP for the local group.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**show running-config** **Example:** `Device# show running-config`	Verifies that a default RP is already configured on all PIM devices and the RP in the sparse-mode network. It was previously configured with the **ip pim rp-address** global configuration command. **Note** This step is not required for spare-dense-mode environments. The selected RP should have good connectivity and be available across the network. Use this RP for the global groups (for example, 224.x.x.x and other global groups). Do not reconfigure the group address range that this RP serves. RPs dynamically discovered through Auto-RP take precedence over statically configured RPs. Assume that it is desirable to use a second RP for the local groups.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 4	**ip pim send-rp-announce** *interface-id* **scope** *ttl* **group-list** *access-list-number* **interval** *seconds* **Example:** `Device(config)# ip pim send-rp-announce gigabitethernet 1/1 scope 20 group-list 10 interval 120`	Configures another PIM device to be the candidate RP for local groups. • For *interface-id*, enter the interface type and number that identifies the RP address. Valid interfaces include physical ports, port channels, and VLANs. • For **scope** *ttl*, specify the time-to-live value in hops. Enter a hop count that is high enough so that the RP-announce messages reach all mapping agents in the network. There is no default setting. The range is 1 to 255. • For **group-list** *access-list-number*, enter an IP standard access list number from 1 to 99. If no access list is configured, the RP is used for all groups. • For **interval** *seconds*, specify how often the announcement messages must be sent. The default is 60 seconds. The range is 1 to 16383.
Step 5	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] **Example:** `Device(config)# access-list 10 permit 10.10.0.0`	Creates a standard access list, repeating the command as many times as necessary. • For *access-list-number*, enter the access list number specified in Step 3. • The **deny** keyword denies access if the conditions are matched. • The **permit** keyword permits access if the conditions are matched. • For *source*, enter the multicast group address range for which the RP should be used. • (Optional) For *source-wildcard*, enter the wildcard bits in dotted decimal notation to be applied to the source. Place ones in the bit positions that you want to ignore. **Note** Recall that the access list is always terminated by an implicit deny statement for everything.

	Command or Action	Purpose
Step 6	**ip pim send-rp-discovery scope** *ttl* **Example:** `Device(config)# ip pim send-rp-discovery scope 50`	Finds a device whose connectivity is not likely to be interrupted, and assign it the role of RP-mapping agent. For **scope** *ttl*, specify the time-to-live value in hops to limit the RP discovery packets. All devices within the hop count from the source device receive the Auto-RP discovery messages. These messages tell other devices which group-to-RP mapping to use to avoid conflicts (such as overlapping group-to-RP ranges). There is no default setting. The range is 1 to 255.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 9	**show ip pim rp mapping** **Example:** `Device# show ip pim rp mapping`	Displays active RPs that are cached with associated multicast routing entries.
Step 10	**show ip pim rp** **Example:** `Device# show ip pim rp`	Displays the information cached in the routing table.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Adding Auto-RP to an Existing Sparse-Mode Cloud

This section contains suggestions for the initial deployment of Auto-RP into an existing sparse-mode cloud to minimize disruption of the existing multicast infrastructure.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**show running-config** **Example:** `Device# show running-config`	Verifies that a default RP is already configured on all PIM devices and the RP in the sparse-mode network. It was previously configured with the **ip pim rp-address** global configuration command. **Note** This step is not required for spare-dense-mode environments. The selected RP should have good connectivity and be available across the network. Use this RP for the global groups (for example, 224.x.x.x and other global groups). Do not reconfigure the group address range that this RP serves. RPs dynamically discovered through Auto-RP take precedence over statically configured RPs. Assume that it is desirable to use a second RP for the local groups.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**ip pim send-rp-announce** *interface-id* **scope** *ttl* **group-list** *access-list-number* **interval** *seconds* **Example:** `Device(config)# ip pim send-rp-announce gigabitethernet 1/5 scope 20 group-list 10 interval 120`	Configures another PIM device to be the candidate RP for local groups. • For *interface-id*, enter the interface type and number that identifies the RP address. Valid interfaces include physical ports, port channels, and VLANs. • For **scope** *ttl*, specify the time-to-live value in hops. Enter a hop count that is high enough so that the RP-announce messages reach all mapping agents in the network. There is no default setting. The range is 1 to 255.

	Command or Action	Purpose
		• For **group-list** *access-list-number*, enter an IP standard access list number from 1 to 99. If no access list is configured, the RP is used for all groups. • For **interval** *seconds*, specify how often the announcement messages must be sent. The default is 60 seconds. The range is 1 to 16383.
Step 5	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] **Example:** `Device(config)# access-list 10 permit 224.0.0.0 15.255.255.255`	Creates a standard access list, repeating the command as many times as necessary. • For *access-list-number*, enter the access list number specified in Step 3. • The **deny** keyword denies access if the conditions are matched. • The **permit** keyword permits access if the conditions are matched. • For *source*, enter the multicast group address range for which the RP should be used. • (Optional) For *source-wildcard*, enter the wildcard bits in dotted decimal notation to be applied to the source. Place ones in the bit positions that you want to ignore. Recall that the access list is always terminated by an implicit deny statement for everything.
Step 6	**ip pim send-rp-discovery scope** *ttl* **Example:** `Device(config)# ip pim send-rp-discovery scope 50`	Finds a device whose connectivity is not likely to be interrupted, and assigns it the role of RP-mapping agent. For **scope** *ttl*, specify the time-to-live value in hops to limit the RP discovery packets. All devices within the hop count from the source device receive the Auto-RP discovery messages. These messages tell other devices which group-to-RP mapping to use to avoid conflicts (such as overlapping group-to-RP ranges). There is no default setting. The range is 1 to 255. **Note** To remove the device as the RP-mapping agent, use the **no ip pim send-rp-discovery** global configuration command.

	Command or Action	Purpose
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 9	**show ip pim rp mapping** **Example:** `Device# show ip pim rp mapping`	Displays active RPs that are cached with associated multicast routing entries.
Step 10	**show ip pim rp** **Example:** `Device# show ip pim rp`	Displays the information cached in the routing table.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Preventing Join Messages to False RPs

Determine whether the **ip pim accept-rp** command was previously configured throughout the network by using the **show running-config** privileged EXEC command. If the **ip pim accept-rp** command is not configured on any device, this problem can be addressed later. In those routers or multilayer switches already configured with the **ip pim accept-rp** command, you must enter the command again to accept the newly advertised RP.

Filtering Incoming RP Announcement Messages

You can add configuration commands to the mapping agents to prevent a maliciously configured router from masquerading as a candidate RP and causing problems.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip pim rp-announce-filter rp-list** *access-list-number* **group-list** *access-list-number* **Example:** `Device(config)# ip pim rp-announce-filter rp-list 10 group-list 14`	Filters incoming RP announcement messages. Enter this command on each mapping agent in the network. Without this command, all incoming RP-announce messages are accepted by default. For **rp-list** *access-list-number*, configure an access list of candidate RP addresses that, if permitted, is accepted for the group ranges supplied in the **group-list** *access-list-number* variable. If this variable is omitted, the filter applies to all multicast groups. If more than one mapping agent is used, the filters must be consistent across all mapping agents to ensure that no conflicts occur in the group-to-RP mapping information.
Step 4	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] **Example:** `Device(config)# access-list 10 permit 10.8.1.0 255.255.224.0`	Creates a standard access list, repeating the command as many times as necessary. • For *access-list-number*, enter the access list number specified in Step 2. • The **deny** keyword denies access if the conditions are matched. • The **permit** keyword permits access if the conditions are matched. • Create an access list that specifies from which routers and multilayer switches the mapping agent accepts candidate RP announcements (rp-list ACL).

	Command or Action	Purpose
		• Create an access list that specifies the range of multicast groups from which to accept or deny (group-list ACL). • For *source*, enter the multicast group address range for which the RP should be used. • (Optional) For *source-wildcard*, enter the wildcard bits in dotted decimal notation to be applied to the source. Place ones in the bit positions that you want to ignore. The access list is always terminated by an implicit deny statement for everything.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring PIMv2 BSR

The process for configuring PIMv2 BSR may involve the following optional tasks:

- Defining the PIM domain border
- Defining the IP multicast boundary
- Configuring candidate BSRs
- Configuring candidate RPs

Defining the PIM Domain Border

Perform the following steps to configure the PIM domain border. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be configured, and enters interface configuration mode.
Step 4	**ip pim bsr-border** **Example:** `Device(config-if)# ip pim bsr-border`	Defines a PIM bootstrap message boundary for the PIM domain. Enter this command on each interface that connects to other bordering PIM domains. This command instructs the device to neither send nor receive PIMv2 BSR messages on this interface. **Note** To remove the PIM border, use the **no ip pim bsr-border** interface configuration command.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:**	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`Device# copy running-config startup-config`	

Defining the IP Multicast Boundary

You define a multicast boundary to prevent Auto-RP messages from entering the PIM domain. You create an access list to deny packets destined for 224.0.1.39 and 224.0.1.40, which carry Auto-RP information.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *access-list-number* **deny** *source* [*source-wildcard*] **Example:** `Device(config)#` `access-list 12 deny 224.0.1.39` `access-list 12 deny 224.0.1.40`	Creates a standard access list, repeating the command as many times as necessary. • For *access-list-number*, the range is 1 to 99. • The **deny** keyword denies access if the conditions are matched. • For *source*, enter multicast addresses 224.0.1.39 and 224.0.1.40, which carry Auto-RP information. • (Optional) For *source-wildcard*, enter the wildcard bits in dotted decimal notation to be applied to the source. Place ones in the bit positions that you want to ignore. The access list is always terminated by an implicit deny statement for everything.
Step 4	**interface** *interface-id* **Example:**	Specifies the interface to be configured, and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet 1/1`	
Step 5	**ip multicast boundary** *access-list-number* **Example:** `Device(config-if)# ip multicast boundary 12`	Configures the boundary, specifying the access list you created in Step 2.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Candidate BSRs

You can configure one or more candidate BSRs. The devices serving as candidate BSRs should have good connectivity to other devices and be in the backbone portion of the network.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ip pim bsr-candidate** *interface-id hash-mask-length* [*priority*] **Example:** `Device(config)# ip pim bsr-candidate gigabitethernet 1/3 28 100`	Configures your device to be a candidate BSR. • For *interface-id*, enter the interface on this device from which the BSR address is derived to make it a candidate. This interface must be enabled with PIM. Valid interfaces include physical ports, port channels, and VLANs. • For *hash-mask-length*, specify the mask length (32 bits maximum) that is to be ANDed with the group address before the hash function is called. All groups with the same seed hash correspond to the same RP. For example, if this value is 24, only the first 24 bits of the group addresses matter. • (Optional) For *priority*, enter a number from 0 to 255. The BSR with the larger priority is preferred. If the priority values are the same, the device with the highest IP address is selected as the BSR. The default is 0.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring the Candidate RPs

You can configure one or more candidate RPs. Similar to BSRs, the RPs should also have good connectivity to other devices and be in the backbone portion of the network. An RP can serve the entire IP multicast address space or a portion of it. Candidate RPs send candidate RP advertisements to the BSR.

This procedure is optional.

Before you begin

When deciding which devices should be RPs, consider these options:

- In a network of Cisco routers and multilayer switches where only Auto-RP is used, any device can be configured as an RP.
- In a network that includes only Cisco PIMv2 routers and multilayer switches and with routers from other vendors, any device can be used as an RP.
- In a network of Cisco PIMv1 routers, Cisco PIMv2 routers, and routers from other vendors, configure only Cisco PIMv2 routers and multilayer switches as RPs.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip pim rp-candidate** *interface-id* [**group-list** *access-list-number*] **Example:** `Device(config)# ip pim rp-candidate gigabitethernet 1/5 group-list 10`	Configures your device to be a candidate RP. • For *interface-id*, specify the interface whose associated IP address is advertised as a candidate RP address. Valid interfaces include physical ports, port channels, and VLANs. • (Optional) For **group-list** *access-list-number*, enter an IP standard access list number from 1 to 99. If no group-list is specified, the device is a candidate RP for all groups.
Step 4	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*]	Creates a standard access list, repeating the command as many times as necessary.

	Command or Action	Purpose
	Example: `Device(config)# access-list 10 permit 239.0.0.0 0.255.255.255`	• For *access-list-number*, enter the access list number specified in Step 2. • The **deny** keyword denies access if the conditions are matched. The **permit** keyword permits access if the conditions are matched. • For *source*, enter the number of the network or host from which the packet is being sent. • (Optional) For *source-wildcard*, enter the wildcard bits in dotted decimal notation to be applied to the source. Place ones in the bit positions that you want to ignore. The access list is always terminated by an implicit deny statement for everything.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Sparse Mode with Auto-RP

Before you begin

- All access lists that are needed when Auto-RP is configured should be configured prior to beginning the configuration task.

Note

- If a group has no known RP and the interface is configured to be sparse-dense mode, the interface is treated as if it were in dense mode, and data is flooded over the interface. To avoid this data flooding, configure the Auto-RP listener and then configure the interface as sparse mode.
- When configuring Auto-RP, you must either configure the Auto-RP listener feature (Step 5) and specify sparse mode (Step 7).
- When you configure sparse-dense mode, dense mode failover may result in a network dense-mode flood. To avoid this condition, use PIM sparse mode with the Auto-RP listener feature.

Follow this procedure to configure auto-rendezvous point (Auto-RP). Auto-RP can also be optionally used with anycast RP.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip multicast-routing** **Example:** `Device(config)# ip multicast-routing`	Enables IP multicast routing.
Step 4	Either perform Steps 5 through 7 or perform Steps 6 and 8.	--
Step 5	**interface** *type number* **Example:** `Device(config)# interface Gigabitethernet 1/1`	Selects an interface that is connected to hosts on which PIM can be enabled.
Step 6	**ip pim sparse-mode** **Example:** `Device(config-if)# ip pim sparse-mode`	Enables PIM sparse mode on an interface. When configuring Auto-RP in sparse mode, you must also configure the Auto-RP listener in the next step. • Skip this step if you are configuring sparse-dense mode in Step 8.

	Command or Action	Purpose
Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	Repeat Steps 1 through 9 on all PIM interfaces.	--
Step 9	**ip pim send-rp-announce** {*interface-type interface-number* | *ip-address*} **scope** *ttl-value* [**group-list** *access-list*] [**interval** *seconds*] [**bidir**] **Example:** `Device(config)# ip pim send-rp-announce loopback0 scope 31 group-list 5`	Sends RP announcements out all PIM-enabled interfaces. • Perform this step on the RP device only. • Use the*interface-type*and *interface-number* arguments to define which IP address is to be used as the RP address. • Use the*ip-address* argument to specify a directly connected IP address as the RP address. **Note** If the*ip-address* argument is configured for this command, the RP-announce message will be sourced by the interface to which this IP address is connected (that is, the source address in the IP header of the RP-announce message is the IP address of that interface). • This example shows that the interface is enabled with a maximum of 31 hops. The IP address by which the device wants to be identified as RP is the IP address associated with loopback interface 0. Access list 5 describes the groups for which this device serves as RP.
Step 10	**ip pim send-rp-discovery** [*interface-type interface-number*] **scope** *ttl-value* [**interval** *seconds*] **Example:** `Device(config)# ip pim send-rp-discovery loopback 1 scope 31`	Configures the device to be an RP mapping agent. • Perform this step on RP mapping agent devices or on combined RP/RP mapping agent devices. **Note** Auto-RP allows the RP function to run separately on one device and the RP mapping agent to run on one or multiple devices. It is possible to deploy the RP and the RP mapping agent on a combined RP/RP mapping agent device.

	Command or Action	Purpose
		• Use the optional *interface-type* and *interface-number* arguments to define which IP address is to be used as the source address of the RP mapping agent. • Use the **scope** keyword and *ttl-value* argument to specify the Time-to-Live (TTL) value in the IP header of Auto-RP discovery messages. • Use the optional **interval** keyword and *seconds* argument to specify the interval at which Auto-RP discovery messages are sent. **Note** Lowering the interval at which Auto-RP discovery messages are sent from the default value of 60 seconds results in more frequent floodings of the group-to-RP mappings. In some network environments, the disadvantages of lowering the interval (more control packet overhead) may outweigh the advantages (more frequent group-to-RP mapping updates). • The example shows limiting the Auto-RP discovery messages to 31 hops on loopback interface 1.
Step 11	**ip pim rp-announce-filter rp-list** *access-list* **group-list** *access-list* **Example:** `Device(config)# ip pim rp-announce-filter rp-list 1 group-list 2`	Filters incoming RP announcement messages sent from candidate RPs (C-RPs) to the RP mapping agent. • Perform this step on the RP mapping agent only.
Step 12	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Selects an interface that is connected to hosts on which PIM can be enabled.
Step 13	**ip multicast boundary** *access-list* [**filter-autorp**] **Example:** `Device(config-if)# ip multicast boundary 10 filter-autorp`	Configures an administratively scoped boundary. • Perform this step on the interfaces that are boundaries to other devices. • The access list is not shown in this task.

	Command or Action	Purpose
		• An access list entry that uses the **deny** keyword creates a multicast boundary for packets that match that entry.
Step 14	**end** **Example:** `Device(config-if)# end`	Returns to global configuration mode.
Step 15	**show ip pim autorp** **Example:** `Device# show ip pim autorp`	(Optional) Displays the Auto-RP information.
Step 16	**show ip pim rp** [**mapping**] [*rp-address*] **Example:** `Device# show ip pim rp mapping`	(Optional) Displays RPs known in the network and shows how the device learned about each RP.
Step 17	**show ip igmp groups** [*group-name* | *group-address*| *interface-type interface-number*] [**detail**] **Example:** `Device# show ip igmp groups`	(Optional) Displays the multicast groups having receivers that are directly connected to the device and that were learned through Internet Group Management Protocol (IGMP). • A receiver must be active on the network at the time that this command is issued in order for receiver information to be present on the resulting display.
Step 18	**show ip mroute** [*group-address* | *group-name*] [*source-address* | *source-name*] [*interface-type interface-number*] [**summary**] [**count**] [**active** *kbps*] **Example:** `Device# show ip mroute cbone-audio`	(Optional) Displays the contents of the IP multicast routing (mroute) table.

Delaying the Use of PIM Shortest-Path Tree

Perform these steps to configure a traffic rate threshold that must be reached before multicast routing is switched from the source tree to the shortest-path tree.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] **Example:** `Device(config)# access-list 16 permit 225.0.0.0 0.255.255.255`	Creates a standard access list. • For *access-list-number*, the range is 1 to 99. • The **deny** keyword denies access if the conditions are matched. • The **permit** keyword permits access if the conditions are matched. • For *source*, specify the multicast group to which the threshold will apply. • (Optional) For *source-wildcard*, enter the wildcard bits in dotted decimal notation to be applied to the source. Place ones in the bit positions that you want to ignore. The access list is always terminated by an implicit deny statement for everything.
Step 4	**ip pim spt-threshold** {*kbps* | **infinity**} [**group-list** *access-list-number*] **Example:** `Device(config)# ip pim spt-threshold infinity group-list 16`	Specifies the threshold that must be reached before moving to shortest-path tree (spt). • For *kbps*, specify the traffic rate in kilobits per second. The default is 0 kbps. **Note** Because of device hardware limitations, 0 kbps is the only valid entry even though the range is 0 to 4294967. • Specify **infinity** if you want all sources for the specified group to use the shared tree, never switching to the source tree.

	Command or Action	Purpose
		• (Optional) For **group-list** *access-list-number*, specify the access list created in Step 2. If the value is 0 or if the group list is not used, the threshold applies to all groups.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Modifying the PIM Router-Query Message Interval

PIM routers and multilayer switches send PIM router-query messages to find which device will be the designated router (DR) for each LAN segment (subnet). The DR is responsible for sending IGMP host-query messages to all hosts on the directly connected LAN.

With PIM DM operation, the DR has meaning only if IGMPv1 is in use. IGMPv1 does not have an IGMP querier election process, so the elected DR functions as the IGMP querier. With PIM-SM operation, the DR is the device that is directly connected to the multicast source. It sends PIM register messages to notify the RP that multicast traffic from a source needs to be forwarded down the shared tree. In this case, the DR is the device with the highest IP address.

This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be configured, and enters interface configuration mode.
Step 4	**ip pim query-interval** *seconds* **Example:** `Device(config-if)# ip pim query-interval 45`	Configures the frequency at which the device sends PIM router-query messages. The default is 30 seconds. The range is 1 to 65535.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show ip igmp interface** [*interface-id*] **Example:** `Device# show ip igmp interface`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Verifying PIM Operations

Verifying IP Multicast Operation in a PIM-SM or a PIM-SSM Network

When you verify the operation of IP multicast in a PIM-SM network environment or in an PIM-SSM network environment, a useful approach is to begin the verification process on the last hop router, and then continue the verification process on the routers along the SPT until the first hop router has been reached. The goal of the verification is to ensure that IP multicast traffic is being routed properly through an IP multicast network.

Perform the following optional tasks to verify IP multicast operation in a PIM-SM or a PIM-SSM network. The steps in these tasks help to locate a faulty hop when sources and receivers are not operating as expected.

Note If packets are not reaching their expected destinations, you might want consider disabling IP multicast fast switching, which would place the router in process switching mode. If packets begin reaching their proper destinations after IP multicast fast switching has been disabled, then the issue most likely was related to IP multicast fast switching.

Verifying IP Multicast on the First Hop Router

Enter these commands on the first hop router to verify IP multicast operations on the first hop router:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip mroute** [*group-address*] **Example:** `Device# show ip mroute 239.1.2.3` `(*, 239.1.2.3), 00:18:10/stopped, RP 172.16.0.1, flags: SPF` `  Incoming interface: Serial1/0, RPF nbr 172.31.200.2` `  Outgoing interface list: Null` `(10.0.0.1, 239.1.2.3), 00:18:10/00:03:22, flags: FT` `  Incoming interface: GigabitEthernet1/1, RPF nbr 0.0.0.0` `  Outgoing interface list:` `  Serial1/0, Forward/Sparse, 00:18:10/00:03:19`	Confirms that the F flag has been set for mroutes on the first hop router.
Step 3	**show ip mroute active** [*kb/s*] **Example:** `Device# show ip mroute active` `Active IP Multicast Sources - sending >= 4 kbps` `Group: 239.1.2.3, (?)` `  Source: 10.0.0.1 (?)` `  Rate: 20 pps/4 kbps(1sec), 4 kbps(last 30 secs), 4 kbps(life avg)`	Displays information about active multicast sources sending to groups. The output of this command provides information about the multicast packet rate for active sources. **Note** By default, the output of the **show ip mroute** command with the **active** keyword displays information about active sources sending traffic to groups at a rate greater than or equal to 4 kb/s. To display information about active

	Command or Action	Purpose
		sources sending low-rate traffic to groups (that is, traffic less than 4 kb/s), specify a value of 1 for the *kb/s* argument. Specifying a value of 1 for this argument displays information about active sources sending traffic to groups at a rate equal to or greater than 1 kb/s, which effectively displays information about all possible active source traffic.

Verifying IP Multicast on Routers Along the SPT

Enter these commands on routers along the SPT to verify IP multicast operations on routers along the SPT in a PIM-SM or PIM-SSM network:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip mroute** [*group-address*] **Example:** `Device# show ip mroute 239.1.2.3` `(*, 239.1.2.3), 00:17:56/00:03:02, RP 172.16.0.1, flags: S` `  Incoming interface: Null, RPF nbr 0.0.0.0` `  Outgoing interface list:` `  GigabitEthernet1/1, Forward/Sparse, 00:17:56/00:03:02` `(10.0.0.1, 239.1.2.3), 00:15:34/00:03:28, flags: T` `  Incoming interface: Serial1/0, RPF nbr 172.31.200.1` `  Outgoing interface list:` `  GigabitEthernet1/1, Forward/Sparse, 00:15:34/00:03:02`	Confirms the RPF neighbor towards the source for a particular group or groups.
Step 3	**show ip mroute active** **Example:** `Device# show ip mroute active` `Active IP Multicast Sources - sending >= 4 kbps` `Group: 239.1.2.3, (?)` `  Source: 10.0.0.1 (?)`	Displays information about active multicast sources sending to groups. The output of this command provides information about the multicast packet rate for active sources. **Note** By default, the output of the **show ip mroute** command with the **active** keyword displays

	Command or Action	Purpose
	`Rate: 20 pps/4 kbps(1sec), 4 kbps(last 30 secs), 4 kbps(life avg)`	information about active sources sending traffic to groups at a rate greater than or equal to 4 kb/s. To display information about active sources sending low-rate traffic to groups (that is, traffic less than 4 kb/s), specify a value of 1 for the *kb/s* argument. Specifying a value of 1 for this argument displays information about active sources sending traffic to groups at a rate equal to or greater than 1 kb/s, which effectively displays information about all possible active source traffic.

Verifying IP Multicast Operation on the Last Hop Router

Enter these commands on the last hop router to verify IP multicast operations on the last hop router:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip igmp groups** **Example:** `Device# show ip igmp groups` `IGMP Connected Group Membership` `Group Address Interface Uptime Expires Last Reporter` `239.1.2.3 GigabitEthernet1/1 00:05:14 00:02:14 10.1.0.6` `224.0.1.39 GigabitEthernet1/2 00:09:11 00:02:08 172.31.100.1`	Verifies IGMP memberships on the last hop router. This information will confirm the multicast groups with receivers that are directly connected to the last hop router and that are learned through IGMP.
Step 3	**show ip pim rp mapping** **Example:** `Device# show ip pim rp mapping` `PIM Group-to-RP Mappings` `Group(s) 224.0.0.0/4` `RP 172.16.0.1 (?), v2v1` `Info source: 172.16.0.1 (?), elected via Auto-RP` `Uptime: 00:09:11, expires: 00:02:47`	Confirms that the group-to-RP mappings are being populated correctly on the last hop router. **Note** Ignore this step if you are verifying a last hop router in a PIM-SSM network. The **show ip pim rp mapping**command does not work with routers in a PIM-SSM network because PIM-SSM does not use RPs. In addition, if configured correctly, PIM-SSM groups do not appear in the output of the **show ip pim rp mapping**command.

	Command or Action	Purpose
Step 4	**show ip mroute** **Example:** `Device# show ip mroute` `(*, 239.1.2.3), 00:05:14/00:03:04, RP 172.16.0.1, flags: SJC` `  Incoming interface: GigabitEthernet1/1, RPF nbr 172.31.100.1` `  Outgoing interface list:` `  GigabitEthernet1/1, Forward/Sparse, 00:05:10/00:03:04` `(10.0.0.1, 239.1.2.3), 00:02:49/00:03:29, flags: T` `  Incoming interface: GigabitEthernet1/2, RPF nbr 172.31.100.1` `  Outgoing interface list:` `  GigabitEthernet1/0, Forward/Sparse, 00:02:49/00:03:04` `(172.16.0.1, 224.0.1.39), 00:02:00/00:01:33, flags: PTX` `  Incoming interface: GigabitEthernet1/3, RPF nbr 172.31.100.1`	Verifies that the mroute table is being populated properly on the last hop router.
Step 5	**show ip interface** [*type number*] **Example:** `Device# show ip interface GigabitEthernet 1/1` `GigabitEthernet0/1 is up, line protocol is up` `  Internet address is 172.31.100.2/24` `  Broadcast address is 255.255.255.255` `  Address determined by setup command` `  MTU is 1500 bytes` `  Helper address is not set` `  Directed broadcast forwarding is disabled` `  Multicast reserved groups joined: 224.0.0.1 224.0.0.22 224.0.0.13` `  224.0.0.5 224.0.0.6` `  Outgoing access list is not set` `  Inbound  access list is not set` `  Proxy ARP is enabled` `  Local Proxy ARP is disabled` `  Security level is default` `  Split horizon is enabled` `  ICMP redirects are always sent` `  ICMP unreachables are always sent` `  ICMP mask replies are never sent` `  IP fast switching is enabled` `  IP fast switching on the same interface is disabled` `  IP Flow switching is disabled` `  IP CEF switching is disabled` `  IP Fast switching turbo vector` `  IP multicast fast switching is enabled` `  IP route-cache flags are Fast` `  Router Discovery is disabled`	Verifies that multicast fast switching is enabled for optimal performance on the outgoing interface on the last hop router. **Note** Using the **no ip mroute-cache** interface command disables IP multicast fast-switching. When IP multicast fast switching is disabled, packets are forwarded through the process-switched path.

	Command or Action	Purpose
	IP output packet accounting is disabled IP access violation accounting is disabled TCP/IP header compression is disabled RTP/IP header compression is disabled Policy routing is disabled Network address translation is disabled WCCP Redirect outbound is disabled WCCP Redirect inbound is disabled WCCP Redirect exclude is disabled BGP Policy Mapping is disabled	
Step 6	**show ip mfib** **Example:** `Device# show ip mfib`	Displays the forwarding entries and interfaces in the IP Multicast Forwarding Information Base (MFIB).
Step 7	**show ip pim interface count** **Example:** `Device# show ip pim interface count` `State: * - Fast Switched, H - Hardware Switching Enabled` `Address Interface FS Mpackets In/Out` `172.31.100.2 GigabitEthernet1/1 * 4122/0` `10.1.0.1 GigabitEthernet1/2 * 0/3193`	Confirms that multicast traffic is being forwarded on the last hop router.
Step 8	**show ip mroute count** **Example:** `Device# show ip mroute count` `IP Multicast Statistics` `6 routes using 4008 bytes of memory` `3 groups, 1.00 average sources per group` `Forwarding Counts: Pkt Count/Pkts per second/Avg Pkt Size/Kilobits per second` `Other counts: Total/RPF failed/Other drops(OIF-null, rate-limit etc)` `Group: 224.0.1.39, Source count: 1, Packets forwarded: 21, Packets received: 120` `Source: 172.16.0.1/32, Forwarding: 21/1/48/0, Other: 120/0/99` `Group: 224.0.1.40, Source count: 1, Packets forwarded: 10, Packets received: 10` `Source: 172.16.0.1/32, Forwarding: 10/1/48/0, Other: 10/0/0`	Confirms that multicast traffic is being forwarded on the last hop router.

	Command or Action	Purpose
Step 9	**show ip mroute active** [*kb/s*] **Example:** `Device# show ip mroute active` `Active IP Multicast Sources - sending >= 4 kbps` `Group: 239.1.2.3, (?)` `Source: 10.0.0.1 (?)` `Rate: 20 pps/4 kbps(1sec), 4 kbps(last 50 secs), 4 kbps(life avg)`	Displays information about active multicast sources sending traffic to groups on the last hop router. The output of this command provides information about the multicast packet rate for active sources. **Note** By default, the output of the **show ip mroute** command with the **active** keyword displays information about active sources sending traffic to groups at a rate greater than or equal to 4 kb/s. To display information about active sources sending low-rate traffic to groups (that is, traffic less than 4 kb/s), specify a value of 1 for the *kb/s* argument. Specifying a value of 1 for this argument displays information about active sources sending traffic to groups at a rate equal to or greater than 1 kb/s, which effectively displays information about all possible active source traffic.

Using PIM-Enabled Routers to Test IP Multicast Reachability

If all the PIM-enabled routers and access servers that you administer are members of a multicast group, pinging that group causes all routers to respond, which can be a useful administrative and debugging tool.

To use PIM-enabled routers to test IP multicast reachability, perform the following tasks:

Configuring Routers to Respond to Multicast Pings

Follow these steps to configure a router to respond to multicast pings. Perform the task on all the interfaces of a router and on all the routers participating in the multicast network:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number*	Enters interface configuration mode.

	Command or Action	Purpose
	Example: `Device(config)# interface gigabitethernet 1/1`	For the *type* and *number* arguments, specify an interface that is directly connected to hosts or is facing hosts.
Step 4	**ip igmp join-group** *group-address* **Example:** `Device(config-if)# ip igmp join-group 225.2.2.2`	(Optional) Configures an interface on the router to join the specified group. For the purpose of this task, configure the same group address for the *group-address* argument on all interfaces on the router participating in the multicast network. **Note** With this method, the router accepts the multicast packets in addition to forwarding them. Accepting the multicast packets prevents the router from fast switching.
Step 5	Repeat Step 3 and Step 4 for each interface on the router participating in the multicast network.	--
Step 6	**end** **Example:** `Device(config-if)# end`	Ends the current configuration session and returns to privileged EXEC mode.

Pinging Routers Configured to Respond to Multicast Pings

Follow these steps on a router to initiate a ping test to the routers configured to respond to multicast pings. This task is used to test IP multicast reachability in a network.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**ping** *group-address* **Example:** `Device# ping 225.2.2.2`	Pings an IP multicast group address. A successful response indicates that the group address is functioning.

Monitoring and Troubleshooting PIM

Monitoring PIM Information

Use the privileged EXEC commands in the following table to monitor your PIM configurations.

Table 76: PIM Monitoring Commands

Command	Purpose
show ip pim all-vrfs tunnel [**tunnel** *tunnel_number* | **verbose**]	Displays all VRFs.
show ip pim autorp	Displays global auto-RP information.
show ip pim boundary	Displays information about mroutes filtered by administratively scoped IPv4 multicast boundaries configured on an interface.
show ip pim interface	Displays information about interfaces configured for Protocol Independent Multicast (PIM).
show ip pim neighbor	Displays the PIM neighbor information.
show ip pim rp[*group-name* | *group-address*]	Displays RP routers associated with a sparse-mode multicast group. This command is available in all software images.
show ip pim tunnel [**tunnel** | **verbose**]	Displays information about Protocol Independent Multicast (PIM) tunnel interfaces
show ip pim vrf { **word** { **all-vrfs** | **autorp** | **boundary** | **bsr-router** | **interface** | **mdt** | **neighbor** | **rp** | **rp-hash** | **tunnel** } }	Displays the VPN routing/forwarding instance.
show ip igmp groups detail	Displays the interested clients that have joined the specific multicast source group.

Monitoring the RP Mapping and BSR Information

Use the privileged EXEC mode in the following table to verify the consistency of group-to-RP mappings:

Table 77: RP Mapping Monitoring Commands

Command	Purpose
show ip pim rp [*hostname* or *IP address* | **mapping** [*hostname* or *IP address* | **elected** | **in-use**] | **metric** [*hostname* or *IP address*]]	Displays all available RP mappings and metrics. This tells you how the device learns of the RP (through the BSR or the Auto-RP mechanism). • (Optional) For the *hostname*, specify the IP name of the group about which to display RPs. • (Optional) For the *IP address*, specify the IP address of the group about which to display RPs. • (Optional) Use the **mapping** keyword to display all group-to-RP mappings of which the Cisco device is aware (either configured or learned from Auto-RP). • (Optional) Use the **metric** keyword to display the RP RPF metric.
show ip pim rp-hash *group*	Displays the RP that was selected for the specified group. That is, on a PIMv2 router or multilayer switch, confirms that the same RP is the one that a PIMv1 system chooses. For *group*, enter the group address for which to display RP information.

Use the privileged EXEC commands in the following table to monitor BSR information:

Table 78: BSR Monitoring Commands

Command	Purpose
show ip pim bsr	Displays information about the elected BSR.
show ip pim bsr-router	Displays information about the BSRv2.

Troubleshooting PIMv1 and PIMv2 Interoperability Problems

When debugging interoperability problems between PIMv1 and PIMv2, check these in the order shown:

1. Verify RP mapping with the **show ip pim rp-hash** privileged EXEC command, making sure that all systems agree on the same RP for the same group.
2. Verify interoperability between different versions of DRs and RPs. Make sure that the RPs are interacting with the DRs properly (by responding with register-stops and forwarding decapsulated data packets from registers).

Configuration Examples for PIM

Example: Enabling PIM Stub Routing

In this example, IP multicast routing is enabled, Switch A PIM uplink port 25 is configured as a routed uplink port with **spare-dense-mode** enabled. PIM stub routing is enabled on the VLAN 100 interfaces and on Gigabit Ethernet port 20.

```
Device(config)# ip multicast-routing
Device(config)# interface GigabitEthernet1/1
Device(config-if)# no switchport
Device(config-if)# ip address 3.1.1.2 255.255.255.0
Device(config-if)# ip pim sparse-dense-mode
Device(config-if)# exit
Device(config)# interface vlan100
Device(config-if)# ip pim passive
Device(config-if)# exit
Device(config)# interface GigabitEthernet1/2
Device(config-if)# ip pim passive
Device(config-if)# exit
Device(config)# interface vlan100
Device(config-if)# ip address 100.1.1.1 255.255.255.0
Device(config-if)# ip pim passive
Device(config-if)# exit
Device(config)# interface GigabitEthernet1/2
Device(config-if)# no switchport
Device(config-if)# ip address 10.1.1.1 255.255.255.0
Device(config-if)# ip pim passive
Device(config-if)# end
```

Example: Verifying PIM Stub Routing

To verify that PIM stub is enabled for each interface, use the **show ip pim interface** command in privileged EXEC mode:

```
Device# show ip pim interface
Address Interface Ver/ Nbr Query DR DR
Mode Count Intvl Prior
3.1.1.2 GigabitEthernet1/1 v2/SD 1 30 1 3.1.1.2
100.1.1.1 Vlan100 v2/P 0 30 1 100.1.1.1
10.1.1.1 GigabitEthernet1/2 v2/P 0 30 1 10.1.1.1
```

Example: Manually Assigning an RP to Multicast Groups

This example shows how to configure the address of the RP to 147.106.6.22 for multicast group 225.2.2.2 only:

```
Device(config)# access-list 1 permit 225.2.2.2 0.0.0.0
Device(config)# ip pim rp-address 147.106.6.22 1
```

Example: Configuring Auto-RP

This example shows how to send RP announcements out all PIM-enabled interfaces for a maximum of 31 hops. The IP address of port 1 is the RP. Access list 5 describes the group for which this device serves as RP:

```
Device(config)# ip pim send-rp-announce gigabitethernet1/1 scope 31 group-list 5
Device(config)# access-list 5 permit 224.0.0.0 15.255.255.255
```

Example: Sparse Mode with Auto-RP

The following example configures sparse mode with Auto-RP:

```
Device(config)# ip multicast-routing
Device(config)# ip pim autorp listener
Device(config)# ip pim send-rp-announce Loopback0 scope 16 group-list 1
Device(config)# ip pim send-rp-discovery Loopback1 scope 16
Device(config)# no ip pim dm-fallback
Device(config)# access-list 1 permit 239.254.2.0 0.0.0.255
Device(config)# access-list 1 permit 239.254.3.0 0.0.0.255
.
.
.
Device(config)# access-list 10 permit 224.0.1.39
Device(config)# access-list 10 permit 224.0.1.40
Device(config)# access-list 10 permit 239.254.2.0 0.0.0.255
Device(config)# access-list 10 permit 239.254.3.0 0.0.0.255
```

Example: Defining the IP Multicast Boundary to Deny Auto-RP Information

This example shows a portion of an IP multicast boundary configuration that denies Auto-RP information:

```
Device(config)# access-list 1 deny 224.0.1.39
Device(config)# access-list 1 deny 224.0.1.40
Device(config)# access-list 1 permit all
Device(config)# interface gigabitethernet1/1
Device(config-if)# ip multicast boundary 1
```

Example: Filtering Incoming RP Announcement Messages

This example shows a sample configuration on an Auto-RP mapping agent that is used to prevent candidate RP announcements from being accepted from unauthorized candidate RPs:

```
Device(config)# ip pim rp-announce-filter rp-list 10 group-list 20
Device(config)# access-list 10 permit host 172.16.5.1
Device(config)# access-list 10 permit host 172.16.2.1
Device(config)# access-list 20 deny 239.0.0.0 0.0.255.255
Device(config)# access-list 20 permit 224.0.0.0 15.255.255.255
```

The mapping agent accepts candidate RP announcements from only two devices, 172.16.5.1 and 172.16.2.1. The mapping agent accepts candidate RP announcements from these two devices only for multicast groups that fall in the group range of 224.0.0.0 to 239.255.255.255. The mapping agent does not accept candidate

RP announcements from any other devices in the network. Furthermore, the mapping agent does not accept candidate RP announcements from 172.16.5.1 or 172.16.2.1 if the announcements are for any groups in the 239.0.0.0 through 239.255.255.255 range. This range is the administratively scoped address range.

Example: Preventing Join Messages to False RPs

If all interfaces are in sparse mode, use a default-configured RP to support the two well-known groups 224.0.1.39 and 224.0.1.40. Auto-RP uses these two well-known groups to collect and distribute RP-mapping information. When this is the case and the **ip pim accept-rp auto-rp** command is configured, another **ip pim accept-rp** command accepting the RP must be configured as follows:

```
Device(config)# ip pim accept-rp 172.10.20.1 1
Device(config)# access-list 1 permit 224.0.1.39
Device(config)# access-list 1 permit 224.0.1.40
```

Example: Configuring Candidate BSRs

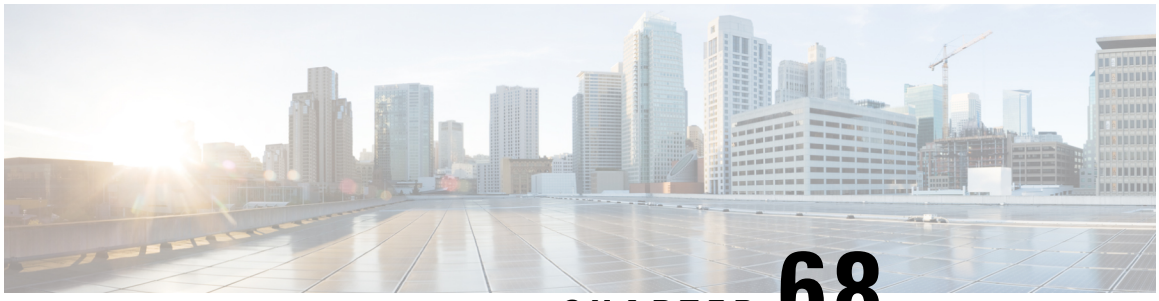
This example shows how to configure a candidate BSR, which uses the IP address 172.21.24.18 on a port as the advertised BSR address, uses 30 bits as the hash-mask-length, and has a priority of 10.

```
Device(config)# interface gigabitethernet1/1
Device(config-if)# ip address 172.21.24.18 255.255.255.0
Device(config-if)# ip pim sparse-mode
Device(config-if)# ip pim bsr-candidate gigabitethernet1/1 30 10
```

Example: Configuring Candidate RPs

This example shows how to configure the device to advertise itself as a candidate RP to the BSR in its PIM domain. Standard access list number 4 specifies the group prefix associated with the RP that has the address identified by a port. That RP is responsible for the groups with the prefix 239.

```
Device(config)# ip pim rp-candidate gigabitethernet1/1 group-list 4
Device(config)# access-list 4 permit 239.0.0.0 0.255.255.255
```



CHAPTER 68

Configuring PIM MIB Extension for IP Multicast

Information About PIM MIB Extension for IP Multicast

PIM MIB Extensions for SNMP Traps for IP Multicast

Protocol Independent Multicast (PIM) is an IP multicast routing protocol used for routing multicast data packets to multicast groups. RFC 2934 defines the PIM MIB for IPv4, which describes managed objects that enable users to remotely monitor and configure PIM using Simple Network Management Protocol (SNMP).

PIM MIB extensions introduce the following new classes of PIM notifications:

- neighbor-change--This notification results from the following conditions:
  - A router’s PIM interface is disabled or enabled (using the **ip pim** command in interface configuration mode)
  - A router's PIM neighbor adjacency expires (defined in RFC 2934)
- rp-mapping-change--This notification results from a change in the rendezvous point (RP) mapping information due to either Auto-RP messages or bootstrap router (BSR) messages.
- invalid-pim-message--This notification results from the following conditions:
  - An invalid (*, G) Join or Prune message is received by the device (for example, when a router receives a Join or Prune message for which the RP specified in the packet is not the RP for the multicast group)
  - An invalid PIM register message is received by the device (for example, when a router receives a register message from a multicast group for which it is not the RP)

Benefits of PIM MIB Extensions

PIM MIB extensions:

- Allow users to identify changes in the multicast topology of their network by detecting changes in the RP mapping.
- Provide traps to monitor the PIM protocol on PIM-enabled interfaces.
- Help users identify routing issues when multicast neighbor adjacencies expire on a multicast interface.
- Enable users to monitor RP configuration errors (for example, errors due to flapping in dynamic RP allocation protocols like Auto-RP).

How to Configure PIM MIB Extension for IP Multicast

Enabling PIM MIB Extensions for IP Multicast

Perform this task to enable PIM MIB extensions for IP multicast.

Note

- The pimInterfaceVersion object was removed from RFC 2934 and, therefore, is no longer supported in software.
- The following MIB tables are not supported in Cisco software:
  - pimIpMRouteTable
  - pimIpMRouteNextHopTable

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server enable traps pim** [**neighbor-change** | **rp-mapping-change** | **invalid-pim-message**] **Example:** `Device(config)# snmp-server enable traps pim neighbor-change`	Enables a device to send PIM notifications. • **neighbor-change** --This keyword enables notifications indicating when a device's PIM interface is disabled or enabled, or when a device's PIM neighbor adjacency expires.

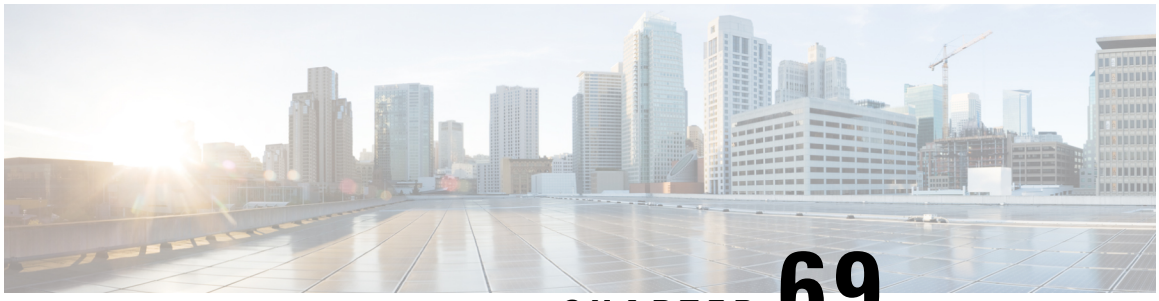
	Command or Action	Purpose
		• **rp-mapping-change** --This keyword enables notifications indicating a change in RP mapping information due to either Auto-RP messages or BSR messages. • **invalid-pim-message** --This keyword enables notifications for monitoring invalid PIM protocol operations (for example, when a device receives a join or prune message for which the RP specified in the packet is not the RP for the multicast group or when a device receives a register message from a multicast group for which it is not the RP).
Step 4	**snmp-server host** *host-address* [**traps** | **informs**] *community-string* **pim** **Example:** `Device(config)# snmp-server host 10.10.10.10 traps public pim`	Specifies the recipient of a PIM SNMP notification operation.

Configuration Examples for PIM MIB Extensions

Example Enabling PIM MIB Extensions for IP Multicast

The following example shows how to configure a router to generate notifications indicating that a PIM interface of the router has been enabled. The first line configures PIM traps to be sent as SNMP v2c traps to the host with IP address 10.0.0.1. The second line configures the router to send the neighbor-change class of trap notification to the host.

```
snmp-server host 10.0.0.1 traps version 2c public pim
snmp-server enable traps pim neighbor-change
interface GigabitEthernet1/1
 ip pim sparse-mode
```



CHAPTER 69

Configuring SSM

Prerequisites for Configuring SSM

The following are the prerequisites for configuring source-specific multicast (SSM) and SSM mapping:

- Before you configure SSM mapping, you must perform the following tasks:
  - Enable IP multicast routing.
  - Enable PIM sparse mode.
  - Configure SSM.
- Before you configure static SSM mapping, you must configure access control lists (ACLs) that define the group ranges to be mapped to source addresses.
- Before you can configure and use SSM mapping with DNS lookups, you need to add records to a running DNS server. If you do not already have a DNS server running, you need to install one.

Note You can use a product such as *Cisco Network Registrar* to add records to a running DNS server.

Restrictions for Configuring SSM

The following are the restrictions for configuring SSM:

- To run SSM with IGMPv3, SSM must be supported in the Cisco IOS router, the host where the application is running, and the application itself.

- Existing applications in a network predating SSM will not work within the SSM range unless they are modified to support (S, G) channel subscriptions. Therefore, enabling SSM in a network may cause problems for existing applications if they use addresses within the designated SSM range.
- IGMP Snooping—IGMPv3 uses new membership report messages that might not be correctly recognized by older IGMP snooping devices.
- Address management is still necessary to some degree when SSM is used with Layer 2 switching mechanisms. Cisco Group Management Protocol (CGMP), IGMP snooping, or Router-Port Group Management Protocol (RGMP) support only group-specific filtering, not (S, G) channel-specific filtering. If different receivers in a switched network request different (S, G) channels sharing the same group, they do not benefit from these existing mechanisms. Instead, both receivers receive all (S, G) channel traffic and filter out the unwanted traffic on input. Because SSM can re-use the group addresses in the SSM range for many independent applications, this situation can lead to decreased traffic filtering in a switched network. For this reason, it is important to use random IP addresses from the SSM range for an application to minimize the chance for re-use of a single address within the SSM range between different applications. For example, an application service providing a set of television channels should, even with SSM, use a different group for each television (S, G) channel. This setup guarantees that multiple receivers to different channels within the same application service never experience traffic aliasing in networks that include Layer 2 devices.
- In PIM-SSM, the last hop router will continue to periodically send (S, G) join messages if appropriate (S, G) subscriptions are on the interfaces. Therefore, as long as receivers send (S, G) subscriptions, the shortest path tree (SPT) state from the receivers to the source will be maintained, even if the source is not sending traffic for longer periods of time (or even never).

  The opposite situation occurs with PIM-SM, where (S, G) state is maintained only if the source is sending traffic and receivers are joining the group. If a source stops sending traffic for more than 3 minutes in PIM-SM, the (S, G) state is deleted and only reestablished after packets from the source arrive again through the RPT (rendezvous point tree). Because no mechanism in PIM-SSM notifies a receiver that a source is active, the network must maintain the (S, G) state in PIM-SSM as long as receivers are requesting receipt of that channel.

The following are the restrictions for configuring SSM mapping:

- The SSM Mapping feature does not share the benefit of full SSM. SSM mapping takes a group G join from a host and identifies this group with an application associated with one or more sources, therefore, it can only support one such application per group G. Nevertheless, full SSM applications may still share the same group also used in SSM mapping.
- Enable IGMPv3 with care on the last hop router when you rely solely on SSM mapping as a transition solution for full SSM. When you enable both SSM mapping and IGMPv3 and the hosts already support IGMPv3 (but not SSM), the hosts send IGMPv3 group reports. SSM mapping does not support these IGMPv3 group reports, and the router does not correctly associate sources with these reports.

Information About SSM

The source-specific multicast (SSM) feature is an extension of IP multicast in which datagram traffic is forwarded to receivers from only those multicast sources that the receivers have explicitly joined. For multicast groups configured for SSM, only SSM distribution trees (no shared trees) are created.

This section describes how to configure source-specific multicast (SSM). For a complete description of the SSM commands in this section, refer to the *IP Multicast Command Reference*.

SSM Components Overview

SSM is a datagram delivery model that best supports one-to-many applications, also known as broadcast applications. SSM is a core networking technology for the Cisco implementation of IP multicast solutions targeted for audio and video broadcast application environments. The device supports the following components that support SSM implementation:

- Protocol independent multicast source-specific mode (PIM-SSM)

  PIM-SSM is the routing protocol that supports the implementation of SSM and is derived from PIM sparse mode (PIM-SM).

- Internet Group Management Protocol version 3 (IGMPv3)

SSM and Internet Standard Multicast (ISM)

The current IP multicast infrastructure in the Internet and many enterprise intranets is based on the PIM-SM protocol and Multicast Source Discovery Protocol (MSDP). These protocols have the limitations of the Internet Standard Multicast (ISM) service model. For example, with ISM, the network must maintain knowledge about which hosts in the network are actively sending multicast traffic.

The ISM service consists of the delivery of IP datagrams from any source to a group of receivers called the multicast host group. The datagram traffic for the multicast host group consists of datagrams with an arbitrary IP unicast source address (S) and the multicast group address (G) as the IP destination address. Systems receive this traffic by becoming members of the host group. Membership in a host group simply requires signaling the host group through IGMP version 1, 2, or 3.

In SSM, delivery of datagrams is based on (S, G) channels. In both SSM and ISM, no signaling is required to become a source. However, in SSM, receivers must subscribe or unsubscribe to (S, G) channels to receive or not receive traffic from specific sources. In other words, receivers can receive traffic only from (S, G) channels to which they are subscribed, whereas in ISM, receivers need not know the IP addresses of sources from which they receive their traffic. The proposed standard approach for channel subscription signaling uses IGMP and includes modes membership reports, which are supported only in IGMP version 3.

SSM IP Address Range

SSM can coexist with the ISM service by applying the SSM delivery model to a configured subset of the IP multicast group address range. Cisco IOS software allows SSM configuration for the IP multicast address range of 224.0.0.0 through 239.255.255.255. When an SSM range is defined, existing IP multicast receiver applications do not receive any traffic when they try to use an address in the SSM range (unless the application is modified to use an explicit (S, G) channel subscription).

SSM Operations

An established network, in which IP multicast service is based on PIM-SM, can support SSM services. SSM can also be deployed alone in a network without the full range of protocols required for interdomain PIM-SM (for example, MSDP, Auto-RP, or bootstrap router [BSR]) if only SSM service is needed.

If SSM is deployed in a network already configured for PIM-SM, only the last-hop routers support SSM. Routers that are not directly connected to receivers do not require support for SSM. In general, these not-last-hop routers must only run PIM-SM in the SSM range and might need additional access control configuration to suppress MSDP signalling, registering, or PIM-SM shared tree operations from occurring within the SSM range.

Use the**ip pim ssm**global configuration command to configure the SSM range and to enable SSM. This configuration has the following effects:

- For groups within the SSM range, (S, G) channel subscriptions are accepted through IGMPv3 include-mode membership reports.
- PIM operations within the SSM range of addresses change to PIM-SSM, a mode derived from PIM-SM. In this mode, only PIM (S, G) join and prune messages are generated by the router, and no (S, G) rendezvous point tree (RPT) or (*, G) RPT messages are generated. Incoming messages related to RPT operations are ignored or rejected, and incoming PIM register messages are immediately answered with register-stop messages. PIM-SSM is backward-compatible with PIM-SM unless a router is a last-hop router. Therefore, routers that are not last-hop routers can run PIM-SM for SSM groups (for example, if they do not yet support SSM).
- No MSDP source-active (SA) messages within the SSM range are accepted, generated, or forwarded.

SSM Mapping

In a typical set-top box (STB) deployment, each TV channel uses one separate IP multicast group and has one active server host sending the TV channel. A single server can send multiple TV channels, but each to a different group. In this network environment, if a router receives an IGMPv1 or IGMPv2 membership report for a particular group, the report addresses the well-known TV server for the TV channel associated with the multicast group.

When SSM mapping is configured, if a router receives an IGMPv1 or IGMPv2 membership report for a particular group, the router translates this report into one or more channel memberships for the well-known sources associated with this group.

When the router receives an IGMPv1 or IGMPv2 membership report for a group, the router uses SSM mapping to determine one or more source IP addresses for the group. SSM mapping then translates the membership report as an IGMPv3 report and continues as if it had received an IGMPv3 report. The router then sends PIM joins and continues to be joined to these groups as long as it continues to receive the IGMPv1 or IGMPv2 membership reports, and the SSM mapping for the group remains the same.

SSM mapping enables the last hop router to determine the source addresses either by a statically configured table on the router or through a DNS server. When the statically configured table or the DNS mapping changes, the router leaves the current sources associated with the joined groups.

Static SSM Mapping

With static SSM mapping, you can configure the last hop router to use a static map to determine the sources that are sending to groups. Static SSM mapping requires that you configure ACLs to define group ranges. After configuring the ACLs to define group ranges, you can then map the groups permitted by those ACLs to sources by using the **ip igmp ssm-map static** global configuration command.

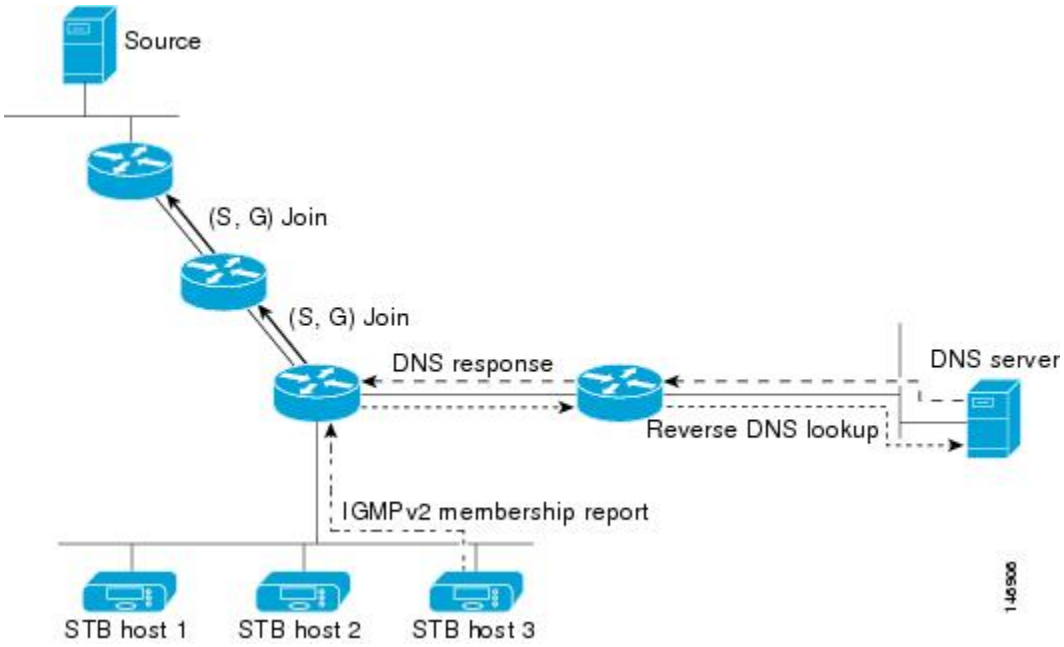
You can configure static SSM mapping in smaller networks when a DNS is not needed or to locally override DNS mappings. When configured, static SSM mappings take precedence over DNS mappings.

DNS-Based SSM Mapping

You can use DNS-based SSM mapping to configure the last hop router to perform a reverse DNS lookup to determine sources sending to groups. When DNS-based SSM mapping is configured, the router constructs a domain name that includes the group address and performs a reverse lookup into the DNS. The router looks up IP address resource records and uses them as the source addresses associated with this group. SSM mapping supports up to 20 sources for each group. The router joins all sources configured for a group.

Figure 78: DNS-Based SSM Mapping

The following figure displays DNS-based SSM mapping.



The SSM mapping mechanism that enables the last hop router to join multiple sources for a group can provide source redundancy for a TV broadcast. In this context, the last hop router provides redundancy using SSM mapping to simultaneously join two video sources for the same TV channel. However, to prevent the last hop router from duplicating the video traffic, the video sources must use a server-side switchover mechanism. One video source is active, and the other backup video source is passive. The passive source waits until an active source failure is detected before sending the video traffic for the TV channel. Thus, the server-side switchover mechanism ensures that only one of the servers is actively sending video traffic for the TV channel.

To look up one or more source addresses for a group that includes G1, G2, G3, and G4, you must configure these DNS records on the DNS server:

```
G4.G3.G2.G1 [multicast-domain] [timeout] IN A source-address-1
 IN A source-address-2
 IN A source-address-n
```

See your DNS server documentation for more information about configuring DNS resource records.

How to Configure SSM

Configuring SSM

Follow these steps to configure SSM:

This procedure is optional.

Before you begin

If you want to use an access list to define the Source Specific Multicast (SSM) range, configure the access list before you reference the access list in the **ip pim ssm** command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip pim ssm** [**default** | **range** *access-list*] **Example:** `Device(config)# ip pim ssm range 20`	Defines the SSM range of IP multicast addresses.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Selects an interface that is connected to hosts on which IGMPv3 can be enabled, and enters the interface configuration mode.
Step 5	**ip pim** {**sparse-mode** } **Example:** `Device(config-if)# ip pim sparse-mode`	Enables PIM on an interface.

	Command or Action	Purpose
Step 6	**ip igmp version 3** **Example:** `Device(config-if)# ip igmp version 3`	Enables IGMPv3 on this interface. The default version of IGMP is set to Version 2.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Source Specific Multicast Mapping

The Source Specific Multicast (SSM) mapping feature supports SSM transition when supporting SSM on the end system is impossible or unwanted due to administrative or technical reasons. You can use SSM mapping to leverage SSM for video delivery to legacy STBs that do not support IGMPv3 or for applications that do not use the IGMPv3 host stack.

Configuring Static SSM Mapping

Follow these steps to configure static SSM Mapping:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp ssm-map enable** **Example:** `Device(config)# ip igmp ssm-map enable`	Enables SSM mapping for groups in the configured SSM range. **Note** By default, this command enables DNS-based SSM mapping.
Step 4	**no ip igmp ssm-map query dns** **Example:** `Device(config)# no ip igmp ssm-map query dns`	(Optional) Disables DNS-based SSM mapping. **Note** Disable DNS-based SSM mapping if you only want to rely on static SSM mapping. By default, the **ip igmp ssm-map** command enables DNS-based SSM mapping.
Step 5	**ip igmp ssm-map static** *access-list source-address* **Example:** `Device(config)# ip igmp ssm-map static 11 172.16.8.11`	Configures static SSM mapping. • The ACL supplied for the *access-list* argument defines the groups to be mapped to the source IP address entered for the *source-address* argument. **Note** You can configure additional static SSM mappings. If additional SSM mappings are configured and the router receives an IGMPv1 or IGMPv2 membership report for a group in the SSM range, the device determines the source addresses associated with the group by walking each configured **ip igmp ssm-map static** command. The device associates up to 20 sources per group. Repeat Step to configure additional static SSM mappings, if required.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:**	Verifies your entries.

	Command or Action	Purpose
	`Device# show running-config`	
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring DNS-Based SSM Mapping

To configure DNS-based SSM mapping, you need to create a DNS server zone or add records to an existing zone. If the routers that are using DNS-based SSM mapping are also using DNS for other purposes, you should use a normally configured DNS server. If DNS-based SSM mapping is the only DNS implementation being used on the router, you can configure a false DNS setup with an empty root zone or a root zone that points back to itself.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp ssm-map enable** **Example:** `Device(config)# ip igmp ssm-map enable`	Enables SSM mapping for groups in a configured SSM range.
Step 4	**ip igmp ssm-map query dns** **Example:** `Device(config)# ip igmp ssm-map query dns`	(Optional) Enables DNS-based SSM mapping. • By default, the **ip igmp ssm-map** command enables DNS-based SSM mapping. Only the **no** form of this command is saved to the running configuration. **Note**

	Command or Action	Purpose
		Use this command to reenable DNS-based SSM mapping if DNS-based SSM mapping is disabled.
Step 5	**ip domain multicast** *domain-prefix* **Example:** `Device(config)# ip domain multicast ssm-map.cisco.com`	(Optional) Changes the domain prefix used for DNS-based SSM mapping. • By default, the software uses the ip-addr.arpa domain prefix.
Step 6	**ip name-server** *server-address1* [*server-address2...server-address6*] **Example:** `Device(config)# ip name-server 10.48.81.21`	Specifies the address of one or more name servers to use for name and address resolution.
Step 7	Repeat Step 6 to configure additional DNS servers for redundancy, if required.	
Step 8	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 9	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 10	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Static Traffic Forwarding with SSM Mapping

Follow these steps to configure static traffic forwarding with SSM mapping on the last hop router:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Selects an interface on which to statically forward traffic for a multicast group using SSM mapping, and enters interface configuration mode. **Note** Static forwarding of traffic with SSM mapping works with either DNS-based SSM mapping or statically configured SSM mapping.
Step 4	**ip igmp static-group** *group-address* **source ssm-map** **Example:** `Device(config-if)# ip igmp static-group 239.1.2.1 source ssm-map`	Configures SSM mapping to statically forward a (S, G) channel from the interface. Use this command if you want to statically forward SSM traffic for certain groups. Use DNS-based SSM mapping to determine the source addresses of the channels.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring SSM

Use the privileged EXEC commands in the following table to monitor SSM.

Table 79: Commands for Monitoring SSM

Command	Purpose
show ip igmp groups detail	Displays the (S, G) channel subscription through IGMPv3.
show ip mroute	Displays whether a multicast group supports SSM service or whether a source-specific host report was received.

Monitoring SSM Mapping

Use the privileged EXEC commands in the following table to monitor SSM mapping.

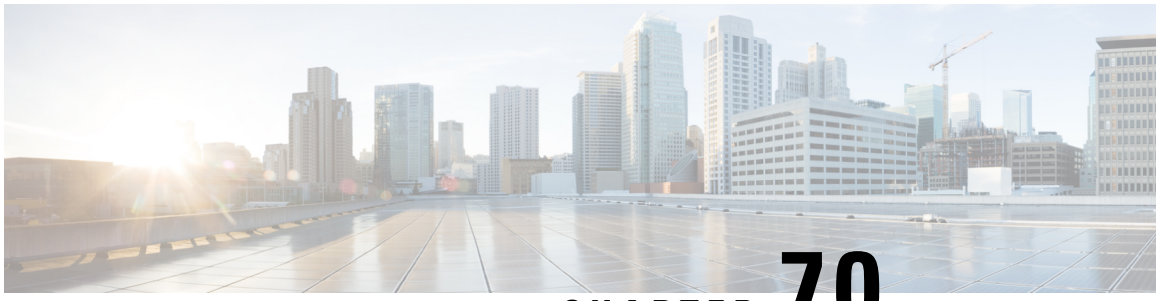
Table 80: SSM Mapping Monitoring Commands

Command	Purpose
show ip igmp ssm-mapping	Displays information about SSM mapping.
show ip igmp ssm-mapping *group-address*	Displays the sources that SSM mapping uses for a particular group.
show ip igmp groups [*group-name* | *group-address* | *interface-type interface-number*] [**detail**]	Displays the multicast groups with receivers that are directly connected to the router and that were learned through IGMP.
show host	Displays the default domain name, the style of name lookup service, a list of name server hosts, and the cached list of hostnames and addresses.
debug ip igmp *group-address*	Displays the IGMP packets received and sent and IGMP host-related events.

Where to Go Next for SSM

You can configure the following:

- IGMP
- PIM
- IP Multicast Routing
- Service Discovery Gateway



CHAPTER 70

Implementing IPv6 Multicast

Information About Implementing IPv6 Multicast Routing

This chapter describes how to implement IPv6 multicast routing on the switch.

Traditional IP communication allows a host to send packets to a single host (unicast transmission) or to all hosts (broadcast transmission). IPv6 multicast provides a third scheme, allowing a host to send a single data stream to a subset of all hosts (group transmission) simultaneously.

IPv6 Multicast Overview

An IPv6 multicast group is an arbitrary group of receivers that want to receive a particular data stream. This group has no physical or geographical boundaries--receivers can be located anywhere on the Internet or in any private network. Receivers that are interested in receiving data flowing to a particular group must join the group by signaling their local switch. This signaling is achieved with the MLD protocol.

Switches use the MLD protocol to learn whether members of a group are present on their directly attached subnets. Hosts join multicast groups by sending MLD report messages. The network then delivers data to a potentially unlimited number of receivers, using only one copy of the multicast data on each subnet. IPv6 hosts that wish to receive the traffic are known as group members.

Packets delivered to group members are identified by a single multicast group address. Multicast packets are delivered to a group using best-effort reliability, just like IPv6 unicast packets.

The multicast environment consists of senders and receivers. Any host, regardless of whether it is a member of a group, can send to a group. However, only members of a group can listen to and receive the message.

A multicast address is chosen for the receivers in a multicast group. Senders use that address as the destination address of a datagram to reach all members of the group.

Note As per RFC 4291, the FF0x::/12 (where the T flag is set to 0 in IPv6 destination address) is for permanently assigned (“well-known”) IPv6 multicast address range.

The default behavior for packets with this address range is to flood in the ingress VLAN.

Membership in a multicast group is dynamic; hosts can join and leave at any time. There is no restriction on the location or number of members in a multicast group. A host can be a member of more than one multicast group at a time.

How active a multicast group is, its duration, and its membership can vary from group to group and from time to time. A group that has members may have no activity.

IPv6 Multicast Routing Implementation

The Cisco IOS software supports the following protocols to implement IPv6 multicast routing:

- MLD is used by IPv6 switches to discover multicast listeners (nodes that want to receive multicast packets destined for specific multicast addresses) on directly attached links. There are two versions of MLD: MLD version 1 is based on version 2 of the Internet Group Management Protocol (IGMP) for IPv4, and MLD version 2 is based on version 3 of the IGMP for IPv4. IPv6 multicast for Cisco IOS software uses both MLD version 2 and MLD version 1. MLD version 2 is fully backward-compatible with MLD version 1 (described in RFC 2710). Hosts that support only MLD version 1 will interoperate with a switch running MLD version 2. Mixed LANs with both MLD version 1 and MLD version 2 hosts are likewise supported.
- PIM-SM is used between switches so that they can track which multicast packets to forward to each other and to their directly connected LANs.
- PIM in Source Specific Multicast (PIM-SSM) is similar to PIM-SM with the additional ability to report interest in receiving packets from specific source addresses (or from all but the specific source addresses) to an IP multicast address.

IPv6 Multicast Listener Discovery Protocol

To start implementing multicasting in the campus network, users must first define who receives the multicast. The MLD protocol is used by IPv6 switches to discover the presence of multicast listeners (for example, nodes that want to receive multicast packets) on their directly attached links, and to discover specifically which multicast addresses are of interest to those neighboring nodes. It is used for discovering local group and source-specific group membership.

The MLD protocol provides a means to automatically control and limit the flow of multicast traffic throughout your network with the use of special multicast queriers and hosts.

Multicast Queriers and Hosts

A multicast querier is a network device, such as a switch, that sends query messages to discover which network devices are members of a given multicast group.

A multicast host is a receiver, including switches, that send report messages to inform the querier of a host membership.

A set of queriers and hosts that receive multicast data streams from the same source is called a multicast group. Queriers and hosts use MLD reports to join and leave multicast groups and to begin receiving group traffic.

MLD uses the Internet Control Message Protocol (ICMP) to carry its messages. All MLD messages are link-local with a hop limit of 1, and they all have the switch alert option set. The switch alert option implies an implementation of the hop-by-hop option header.

MLD Access Group

The MLD access group provides receiver access control in Cisco IOS IPv6 multicast switches. This feature limits the list of groups a receiver can join, and it allows or denies sources used to join SSM channels.

Explicit Tracking of Receivers

The explicit tracking feature allows a switch to track the behavior of the hosts within its IPv6 network. This feature also enables the fast leave mechanism to be used with MLD version 2 host reports.

Protocol Independent Multicast

Protocol Independent Multicast (PIM) is used between switches so that they can track which multicast packets to forward to each other and to their directly connected LANs. PIM works independently of the unicast routing protocol to perform send or receive multicast route updates like other protocols. Regardless of which unicast routing protocols are being used in the LAN to populate the unicast routing table, Cisco IOS PIM uses the existing unicast table content to perform the Reverse Path Forwarding (RPF) check instead of building and maintaining its own separate routing table.

You can configure IPv6 multicast to use either PIM-SM or PIM-SSM operation, or you can use both PIM-SM and PIM-SSM together in your network.

Note IPv6 PIM Dense mode is not supported.

PIM-Sparse Mode

IPv6 multicast provides support for intradomain multicast routing using PIM-SM. PIM-SM uses unicast routing to provide reverse-path information for multicast tree building, but it is not dependent on any particular unicast routing protocol.

PIM-SM is used in a multicast network when relatively few switches are involved in each multicast and these switches do not forward multicast packets for a group, unless there is an explicit request for the traffic. PIM-SM distributes information about active sources by forwarding data packets on the shared tree. PIM-SM initially uses shared trees, which requires the use of an RP.

Requests are accomplished via PIM joins, which are sent hop by hop toward the root node of the tree. The root node of a tree in PIM-SM is the RP in the case of a shared tree or the first-hop switch that is directly connected to the multicast source in the case of a shortest path tree (SPT). The RP keeps track of multicast groups and the hosts that send multicast packets are registered with the RP by that host's first-hop switch.

As a PIM join travels up the tree, switches along the path set up multicast forwarding state so that the requested multicast traffic will be forwarded back down the tree. When multicast traffic is no longer needed, a switch sends a PIM prune up the tree toward the root node to prune (or remove) the unnecessary traffic. As this PIM prune travels hop by hop up the tree, each switch updates its forwarding state appropriately. Ultimately, the forwarding state associated with a multicast group or source is removed.

A multicast data sender sends data destined for a multicast group. The designated switch (DR) of the sender takes those data packets, unicast-encapsulates them, and sends them directly to the RP. The RP receives these encapsulated data packets, de-encapsulates them, and forwards them onto the shared tree. The packets then follow the (*, G) multicast tree state in the switches on the RP tree, being replicated wherever the RP tree

branches, and eventually reaching all the receivers for that multicast group. The process of encapsulating data packets to the RP is called registering, and the encapsulation packets are called PIM register packets.

IPv6 BSR: Configure RP Mapping

PIM switches in a domain must be able to map each multicast group to the correct RP address. The BSR protocol for PIM-SM provides a dynamic, adaptive mechanism to distribute group-to-RP mapping information rapidly throughout a domain. With the IPv6 BSR feature, if an RP becomes unreachable, it will be detected and the mapping tables will be modified so that the unreachable RP is no longer used, and the new tables will be rapidly distributed throughout the domain.

Every PIM-SM multicast group needs to be associated with the IP or IPv6 address of an RP. When a new multicast sender starts sending, its local DR will encapsulate these data packets in a PIM register message and send them to the RP for that multicast group. When a new multicast receiver joins, its local DR will send a PIM join message to the RP for that multicast group. When any PIM switch sends a (*, G) join message, the PIM switch needs to know which is the next switch toward the RP so that G (Group) can send a message to that switch. Also, when a PIM switch is forwarding data packets using (*, G) state, the PIM switch needs to know which is the correct incoming interface for packets destined for G, because it needs to reject any packets that arrive on other interfaces.

A small set of switches from a domain are configured as candidate bootstrap switches (C-BSRs) and a single BSR is selected for that domain. A set of switches within a domain are also configured as candidate RPs (C-RPs); typically, these switches are the same switches that are configured as C-BSRs. Candidate RPs periodically unicast candidate-RP-advertisement (C-RP-Adv) messages to the BSR of that domain, advertising their willingness to be an RP. A C-RP-Adv message includes the address of the advertising C-RP, and an optional list of group addresses and mask length fields, indicating the group prefixes for which the candidacy is advertised. The BSR then includes a set of these C-RPs, along with their corresponding group prefixes, in bootstrap messages (BSMs) it periodically originates. BSMs are distributed hop-by-hop throughout the domain.

Bidirectional BSR support allows bidirectional RPs to be advertised in C-RP messages and bidirectional ranges in the BSM. All switches in a system must be able to use the bidirectional range in the BSM; otherwise, the bidirectional RP feature will not function.

PIM-Source Specific Multicast

PIM-SSM is the routing protocol that supports the implementation of SSM and is derived from PIM-SM. However, unlike PIM-SM where data from all multicast sources are sent when there is a PIM join, the SSM feature forwards datagram traffic to receivers from only those multicast sources that the receivers have explicitly joined, thus optimizing bandwidth utilization and denying unwanted Internet broadcast traffic. Further, instead of the use of RP and shared trees, SSM uses information found on source addresses for a multicast group. This information is provided by receivers through the source addresses relayed to the last-hop switches by MLD membership reports, resulting in shortest-path trees directly to the sources.

In SSM, delivery of datagrams is based on (S, G) channels. Traffic for one (S, G) channel consists of datagrams with an IPv6 unicast source address S and the multicast group address G as the IPv6 destination address. Systems will receive this traffic by becoming members of the (S, G) channel. Signaling is not required, but receivers must subscribe or unsubscribe to (S, G) channels to receive or not receive traffic from specific sources.

MLD version 2 is required for SSM to operate. MLD allows the host to provide source information. Before SSM can run with MLD, SSM must be supported in the Cisco IOS IPv6 switch, the host where the application is running, and the application itself.

Routable Address Hello Option

When an IPv6 interior gateway protocol is used to build the unicast routing table, the procedure to detect the upstream switch address assumes the address of a PIM neighbor is always same as the address of the next-hop switch, as long as they refer to the same switch. However, it may not be the case when a switch has multiple addresses on a link.

Two typical situations can lead to this situation for IPv6. The first situation can occur when the unicast routing table is not built by an IPv6 interior gateway protocol such as multicast BGP. The second situation occurs when the address of an RP shares a subnet prefix with downstream switches (note that the RP switch address has to be domain-wide and therefore cannot be a link-local address).

The routable address hello option allows the PIM protocol to avoid such situations by adding a PIM hello message option that includes all the addresses on the interface on which the PIM hello message is advertised. When a PIM switch finds an upstream switch for some address, the result of RPF calculation is compared with the addresses in this option, in addition to the PIM neighbor's address itself. Because this option includes all the possible addresses of a PIM switch on that link, it always includes the RPF calculation result if it refers to the PIM switch supporting this option.

Because of size restrictions on PIM messages and the requirement that a routable address hello option fits within a single PIM hello message, a limit of 16 addresses can be configured on the interface.

PIM IPv6 Stub Routing

The PIM stub routing feature reduces resource usage by moving routed traffic closer to the end user.

In a network using PIM stub routing, the only allowable route for IPv6 traffic to the user is through a switch that is configured with PIM stub routing. PIM passive interfaces are connected to Layer 2 access domains, such as VLANs, or to interfaces that are connected to other Layer 2 devices. Only directly connected multicast receivers and sources are allowed in the Layer 2 access domains. The PIM passive interfaces do not send or process any received PIM control packets.

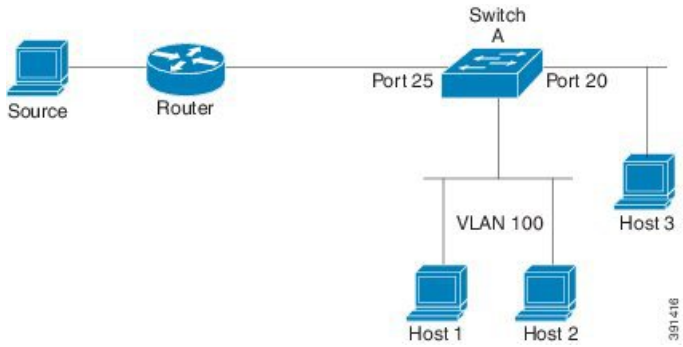
When using PIM stub routing, you should configure the distribution and remote routers to use IPv6 multicast routing and configure only the switch as a PIM stub router. The switch does not route transit traffic between distribution routers. You also need to configure a routed uplink port on the switch. The switch uplink port cannot be used with SVIs.

You must also configure EIGRP stub routing when configuring PIM stub routing on the switch.

The redundant PIM stub router topology is not supported. The redundant topology exists when there is more than one PIM router forwarding multicast traffic to a single access domain. PIM messages are blocked, and the PIM assert and designated router election mechanisms are not supported on the PIM passive interfaces. Only the non-redundant access router topology is supported by the PIM stub feature. By using a non-redundant topology, the PIM passive interface assumes that it is the only interface and designated router on that access domain.

In the figure shown below, Switch A routed uplink port 25 is connected to the router and PIM stub routing is enabled on the VLAN 100 interfaces and on Host 3. This configuration allows the directly connected hosts to receive traffic from multicast source.

Figure 79: PIM Stub Router Configuration



Rendezvous Point

IPv6 PIM provides embedded RP support. Embedded RP support allows the device to learn RP information using the multicast group destination address instead of the statically configured RP. For devices that are the RP, the device must be statically configured as the RP.

The device searches for embedded RP group addresses in MLD reports or PIM messages and data packets. On finding such an address, the device learns the RP for the group from the address itself. It then uses this learned RP for all protocol activity for the group. For devices that are the RP, the device is advertised as an embedded RP must be configured as the RP.

To select a static RP over an embedded RP, the specific embedded RP group range or mask must be configured in the access list of the static RP. When PIM is configured in sparse mode, you must also choose one or more devices to operate as an RP. An RP is a single common root placed at a chosen point of a shared distribution tree and is configured statically in each box.

PIM DRs forward data from directly connected multicast sources to the RP for distribution down the shared tree. Data is forwarded to the RP in one of two ways:

- Data is encapsulated in register packets and unicast directly to the RP by the first-hop device operating as the DR.
- If the RP has itself joined the source tree, it is multicast-forwarded per the RPF forwarding algorithm described in the PIM-Sparse Mode section.

The RP address is used by first-hop devices to send PIM register messages on behalf of a host sending a packet to the group. The RP address is also used by last-hop devices to send PIM join and prune messages to the RP to inform it about group membership. You must configure the RP address on all devices (including the RP device).

A PIM device can be an RP for more than one group. Only one RP address can be used at a time within a PIM domain for a certain group. The conditions specified by the access list determine for which groups the device is an RP.

IPv6 multicast supports the PIM accept register feature, which is the ability to perform PIM-SM register message filtering at the RP. The user can match an access list or compare the AS path for the registered source with the AS path specified in a route map.

Static Mroutes

IPv6 static mroutes behave much in the same way as IPv4 static mroutes used to influence the RPF check. IPv6 static mroutes share the same database as IPv6 static routes and are implemented by extending static route support for RPF checks. Static mroutes support equal-cost multipath mroutes, and they also support unicast-only static routes.

MRIB

The Multicast Routing Information Base (MRIB) is a protocol-independent repository of multicast routing entries instantiated by multicast routing protocols (routing clients). Its main function is to provide independence between routing protocols and the Multicast Forwarding Information Base (MFIB). It also acts as a coordination and communication point among its clients.

Routing clients use the services provided by the MRIB to instantiate routing entries and retrieve changes made to routing entries by other clients. Besides routing clients, MRIB also has forwarding clients (MFIB instances) and special clients such as MLD. MFIB retrieves its forwarding entries from MRIB and notifies the MRIB of any events related to packet reception. These notifications can either be explicitly requested by routing clients or spontaneously generated by the MFIB.

Another important function of the MRIB is to allow for the coordination of multiple routing clients in establishing multicast connectivity within the same multicast session. MRIB also allows for the coordination between MLD and routing protocols.

MFIB

The MFIB is a platform-independent and routing-protocol-independent library for IPv6 software. Its main purpose is to provide a Cisco IOS platform with an interface with which to read the IPv6 multicast forwarding table and notifications when the forwarding table changes. The information provided by the MFIB has clearly defined forwarding semantics and is designed to make it easy for the platform to translate to its specific hardware or software forwarding mechanisms.

When routing or topology changes occur in the network, the IPv6 routing table is updated, and those changes are reflected in the MFIB. The MFIB maintains next-hop address information based on the information in the IPv6 routing table. Because there is a one-to-one correlation between MFIB entries and routing table entries, the MFIB contains all known routes and eliminates the need for route cache maintenance that is associated with switching paths such as fast switching and optimum switching.

Distributed MFIB

Distributed MFIB (dMFIB) is used to switch multicast IPv6 packets on distributed platforms. dMFIB may also contain platform-specific information on replication across line cards. The basic MFIB routines that implement the core of the forwarding logic are common to all forwarding environments.

dMFIB implements the following functions:

- Relays data-driven protocol events generated in the line cards to PIM.
- Provides an MFIB platform application program interface (API) to propagate MFIB changes to platform-specific code responsible for programming the hardware acceleration engine. This API also includes entry points to switch a packet in software (necessary if the packet is triggering a data-driven event) and to upload traffic statistics to the software.

The combination of MFIB and MRIB subsystems also allows the switch to have a "customized" copy of the MFIB database in each line card and to transport MFIB-related platform-specific information from the RP to the line cards.

IPv6 Multicast VRF Lite

Note The switch must be running the Network Advantage license to support IPv6 Multicast VRF lite.

The IPv6 Multicast VRF Lite feature provides IPv6 multicast support for multiple virtual routing/forwarding contexts (VRFs). The scope of these VRFs is limited to the device in which the VRFs are defined.

This feature provides separation between routing and forwarding, providing an additional level of security because no communication between devices belonging to different VRFs is allowed unless it is explicitly configured. The IPv6 Multicast VRF Lite feature simplifies the management and troubleshooting of traffic belonging to a specific VRF.

IPv6 Multicast Process Switching and Fast Switching

A unified MFIB is used to provide both fast switching and process switching support for PIM-SM and PIM-SSM in IPv6 multicast. In process switching, the must examine, rewrite, and forward each packet. The packet is first received and copied into the system memory. The switch then looks up the Layer 3 network address in the routing table. The Layer 2 frame is then rewritten with the next-hop destination address and sent to the outgoing interface. The also computes the cyclic redundancy check (CRC). This switching method is the least scalable method for switching IPv6 packets.

IPv6 multicast fast switching allows switches to provide better packet forwarding performance than process switching. Information conventionally stored in a route cache is stored in several data structures for IPv6 multicast switching. The data structures provide optimized lookup for efficient packet forwarding.

In IPv6 multicast forwarding, the first packet is fast-switched if the PIM protocol logic allows it. In IPv6 multicast fast switching, the MAC encapsulation header is precomputed. IPv6 multicast fast switching uses the MFIB to make IPv6 destination prefix-based switching decisions. In addition to the MFIB, IPv6 multicast fast switching uses adjacency tables to prepend Layer 2 addressing information. The adjacency table maintains Layer 2 next-hop addresses for all MFIB entries.

The adjacency table is populated as adjacencies are discovered. Each time an adjacency entry is created (such as through ARP), a link-layer header for that adjacent node is precomputed and stored in the adjacency table. Once a route is determined, it points to a next hop and corresponding adjacency entry. It is subsequently used for encapsulation during switching of packets.

A route might have several paths to a destination prefix, such as when a switch is configured for simultaneous load balancing and redundancy. For each resolved path, a pointer is added for the adjacency corresponding to the next-hop interface for that path. This mechanism is used for load balancing across several paths.

NTP in IPv6

The Network Time Protocol (NTP) is a protocol designed to time-synchronize a network of machines. NTP runs over UDP, which in turn runs over IPv4. NTP Version 4 (NTPv4) is an extension of NTP version 3, which supports both IPv4 and IPv6.

For more information, see *Cisco IOS IPv6 Configuration Library* on Cisco.com.

How to Implement IPv6 Multicast

Enabling IPv6 Multicast Routing

To enable IPv6 multicast routing, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enter global configuration mode.
Step 3	**ipv6 multicast-routing** **Example:** `Device(config)# ipv6 multicast-routing`	Enables multicast routing on all IPv6-enabled interfaces and enables multicast forwarding for PIM and MLD on all enabled interfaces of the switch.
Step 4	**copy running-config startup-config** **Example:** `Device(config)# copy running-config startup-config`	(Optional) Save your entries in the configuration file.

Customizing and Verifying the MLD Protocol

Customizing and Verifying MLD on an Interface

To customize and verify MLD on an interface, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies an interface type and number, and places the switch in interface configuration mode.
Step 4	**ipv6 mld join-group** [*group-address*] [**include** | **exclude**] {*source-address* | **source-list** [*acl*]} **Example:** `Device(config-if)# ipv6 mld join-group FF04::10`	Configures MLD reporting for a specified group and source.
Step 5	**ipv6 mld access-group** *access-list-name* **Example:** `Device(config-if)# ipv6 access-list acc-grp-1`	Allows the user to perform IPv6 multicast receiver access control.
Step 6	**ipv6 mld static-group** [*group-address*] [**include** | **exclude**] {*source-address* | source-list [*acl*]} **Example:** `Device(config-if)# ipv6 mld static-group ff04::10 include 100::1`	Statically forwards traffic for the multicast group onto a specified interface and cause the interface to behave as if a MLD joiner were present on the interface.
Step 7	**ipv6 mld query-max-response-time** *seconds* **Example:** `Device(config-if)# ipv6 mld query-timeout 130`	Configures the timeout value before the switch takes over as the querier for the interface.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Enter this command twice to exit interface configuration mode and enter privileged EXEC mode.

	Command or Action	Purpose
Step 9	**show ipv6 mld groups [link-local]** [*group-name* | *group-address*] [*interface-type interface-number*] [**detail** | **explicit**] **Example:** `Device# show ipv6 mld groups GigabitEthernet 1/1`	Displays the multicast groups that are directly connected to the switch and that were learned through MLD.
Step 10	**show ipv6 mld groups summary** **Example:** `Device# show ipv6 mld groups summary`	Displays the number of (*, G) and (S, G) membership reports present in the MLD cache.
Step 11	**show ipv6 mld interface** [*type number*] **Example:** `Device# show ipv6 mld interface GigabitEthernet 1/1`	Displays multicast-related information about an interface.
Step 12	**debug ipv6 mld** [*group-name* | *group-address* | *interface-type*] **Example:** `Device# debug ipv6 mld`	Enables debugging on MLD protocol activity.
Step 13	**debug ipv6 mld explicit** [*group-name* | *group-address* **Example:** `Device# debug ipv6 mld explicit`	Displays information related to the explicit tracking of hosts.
Step 14	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Implementing MLD Group Limits

Per-interface and global MLD limits operate independently of each other. Both per-interface and global MLD limits can be configured on the same switch. The number of MLD limits, globally or per interface, is not configured by default; the limits must be configured by the user. A membership report that exceeds either the per-interface or the global state limit is ignored.

Implementing MLD Group Limits Globally

To implement MLD group limits globally, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mld** [*vrf vrf-name*] **state-limit** *number* **Example:** `Device(config)# ipv6 mld state-limit 300`	Limits the number of MLD states globally.
Step 4	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Implementing MLD Group Limits per Interface

To implement MLD group limits per interface, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface type** *number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies an interface type and number, and places the switch in interface configuration mode.

	Command or Action	Purpose
Step 4	**ipv6 mld limit** *number* [**except**]*access-list* **Example:** `Device(config-if)# ipv6 mld limit 100`	Limits the number of MLD states on a per-interface basis.
Step 5	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Configuring Explicit Tracking of Receivers to Track Host Behavior

The explicit tracking feature allows a switch to track the behavior of the hosts within its IPv6 network and enables the fast leave mechanism to be used with MLD version 2 host reports.

To configuring explicit tracking of receivers to track host behavior, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enter global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies an interface type and number, and places the switch in interface configuration mode.
Step 4	**ipv6 mld explicit-tracking** *access-list-name* **Example:** `Device(config-if)# ipv6 mld explicit-tracking list1`	Enables explicit tracking of hosts.
Step 5	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Resetting the MLD Traffic Counters

To reset the MLD traffic counters, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**clear ipv6 mld traffic** **Example:** `Device# clear ipv6 mld traffic`	Resets all MLD traffic counters.
Step 4	**show ipv6 mld traffic** **Example:** `Device# show ipv6 mld traffic`	Displays the MLD traffic counters.
Step 5	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Clearing the MLD Interface Counters

To clearing the MLD interface counters, perform this procedure

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**clear ipv6 mld counters** *interface-type* **Example:** `Device# clear ipv6 mld counters GigabitEthernet1/1`	Clears the MLD interface counters.
Step 4	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Configuring PIM

This section explains how to configure PIM.

Configuring PIM-SM and Displaying PIM-SM Information for a Group Range

To configuring PIM-SM and view PIM-SM information for a group range, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	ipv6 pim **rp-address** *ipv6-address*[*group-access-list*] **Example:** `Device(config)# ipv6 pim rp-address 2001:DB8::01:800:200E:8C6C acc-grp-1`	Configures the address of a PIM RP for a particular group range.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode, and returns the switch to privileged EXEC mode.
Step 5	**show ipv6 pim interface** [state-on] [**state-off**] [*type-number*] **Example:**	Displays information about interfaces configured for PIM.

	Command or Action	Purpose
	`Device# show ipv6 pim interface`	
Step 6	**show ipv6 pim group-map** [*group-name* | group-address] | [*group-range* | *group-mask*] [**info-source {bsr** | **default** | **embedded-rp** | **static**}] **Example:** `Device# show ipv6 pim group-map`	Displays an IPv6 multicast group mapping table.
Step 7	**show ipv6 pim neighbor** [**detail**] [*interface-type interface-number* | **count**] **Example:** `Device# show ipv6 pim neighbor`	Displays the PIM neighbors discovered by the Cisco IOS software.
Step 8	**show ipv6 pim range-list** [**config**] [*rp-address* | *rp-name*] **Example:** `Device# show ipv6 pim range-list`	Displays information about IPv6 multicast range lists.
Step 9	**show ipv6 pim tunnel** [*interface-type interface-number*] **Example:** `Device# show ipv6 pim tunnel`	Displays information about the PIM register encapsulation and de-encapsulation tunnels on an interface.
Step 10	**debug ipv6 pim** [*group-name* | *group-address* | **interface** *interface-type* | **bsr** | **group** | **neighbor**] **Example:** `Device# debug ipv6 pim`	Enables debugging on PIM protocol activity.
Step 11	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Configuring PIM Options

To configure PIM options, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim spt-threshold infinity** [**group-list** *access-list-name*] **Example:** `Device(config)# ipv6 pim spt-threshold infinity group-list acc-grp-1`	Configures when a PIM leaf switch joins the SPT for the specified groups.
Step 4	**ipv6 pim accept-register** {**list** *access-list* | **route-map** *map-name*} **Example:** `Device(config)# ipv6 pim accept-register route-map reg-filter`	Accepts or rejects registers at the RP.
Step 5	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies an interface type and number, and places the switch in interface configuration mode.
Step 6	**ipv6 pim dr-priority** *value* **Example:** `Device(config-if)# ipv6 pim dr-priority 3`	Configures the DR priority on a PIM switch.
Step 7	**ipv6 pim hello-interval** *seconds* **Example:** `Device(config-if)# ipv6 pim hello-interval 45`	Configures the frequency of PIM hello messages on an interface.
Step 8	**ipv6 pim join-prune-interval** *seconds* **Example:** `Device(config-if)# ipv6 pim join-prune-interval 75`	Configures periodic join and prune announcement intervals for a specified interface.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config-if)# exit`	Enter this command twice to exit interface configuration mode and enter privileged EXEC mode.
Step 10	**ipv6 pim join-prune statistic** [*interface-type*] **Example:** `Device(config-if)# show ipv6 pim join-prune statistic`	Displays the average join-prune aggregation for the most recently aggregated packets for each interface.
Step 11	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Resetting the PIM Traffic Counters

If PIM malfunctions or in order to verify that the expected number of PIM packets are received and sent, the user can clear PIM traffic counters. Once the traffic counters are cleared, the user can enter the show ipv6 pim traffic command to verify that PIM is functioning correctly and that PIM packets are being received and sent correctly.

To resetting the PIM traffic counters, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**clear ipv6 pim traffic** **Example:** `Device# clear ipv6 pim traffic`	Resets the PIM traffic counters.
Step 4	**show ipv6 pim traffic** **Example:** `Device# show ipv6 pim traffic`	Displays the PIM traffic counters.

	Command or Action	Purpose
Step 5	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Clearing the PIM Topology Table to Reset the MRIB Connection

No configuration is necessary to use the MRIB. However, users may in certain situations want to clear the PIM topology table in order to reset the MRIB connection and verify MRIB information.

To clear the PIM topology table to reset the MRIB connection, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**clear ipv6 pim topology** [*group-name* | *group-address*] **Example:** `Device# clear ipv6 pim topology FF04::10`	Clears the PIM topology table.
Step 4	**show ipv6 mrib client** [**filter**] [**name** {*client-name* | *client-name* **:** **client-id**}] **Example:** `Device# show ipv6 mrib client`	Displays multicast-related information about an interface.
Step 5	**show ipv6 mrib route** {**link-local** | **summary** | [*sourceaddress-or-name* | *] [*groupname-or-address*[*prefix-length*]]] **Example:** `Device# show ipv6 mrib route`	Displays the MRIB route information.
Step 6	**show ipv6 pim topology** [*groupname-or-address* [*sourceaddress-or-name*] | **link-local** | **route-count** [**detail**]]	Displays PIM topology table information for a specific group or all groups.

	Command or Action	Purpose
	Example: `Device# show ipv6 pim topology`	
Step 7	**debug ipv6 mrib client** **Example:** `Device# debug ipv6 mrib client`	Enables debugging on MRIB client management activity.
Step 8	**debug ipv6 mrib io** **Example:** `Device# debug ipv6 mrib io`	Enables debugging on MRIB I/O events.
Step 9	**debug ipv6 mrib proxy** **Example:** `Device# debug ipv6 mrib proxy`	Enables debugging on MRIB proxy activity between the switch processor and line cards on distributed switch platforms.
Step 10	**debug ipv6 mrib route** [*group-name* | *group-address*] **Example:** `Device# debug ipv6 mrib route`	Displays information about MRIB routing entry-related activity.
Step 11	**debug ipv6 mrib table** **Example:** `Device# debug ipv6 mrib table`	Enables debugging on MRIB table management activity.
Step 12	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Configuring PIM IPv6 Stub Routing

The PIM Stub routing feature supports multicast routing between the distribution layer and the access layer. It supports two types of PIM interfaces, uplink PIM interfaces, and PIM passive interfaces. A routed interface configured with the PIM passive mode does not pass or forward PIM control traffic, it only passes and forwards MLD traffic.

PIM IPv6 Stub Routing Configuration Guidelines

- Before configuring PIM stub routing, you must have IPv6 multicast routing configured on both the stub router and the central router. You must also have PIM mode (sparse-mode) configured on the uplink interface of the stub router.
- The PIM stub router does not route the transit traffic between the distribution routers. Unicast (EIGRP) stub routing enforces this behavior. You must configure unicast stub routing to assist the PIM stub router behavior. For more information, see the *EIGRP Stub Routing* section.

- Only directly connected multicast (MLD) receivers and sources are allowed in the Layer 2 access domains. The PIM protocol is not supported in access domains.
- The redundant PIM stub router topology is not supported.

Default IPv6 PIM Routing Configuration

This table displays the default IPv6 PIM routing configuration for the .

Table 81: Default Multicast Routing Configuration

Feature	Default Setting
Multicast routing	Disabled on all interfaces.
PIM version	Version 2.
PIM mode	No mode is defined.
PIM stub routing	None configured.
PIM RP address	None configured.
PIM domain border	Disabled.
PIM multicast boundary	None.
Candidate BSRs	Disabled.
Candidate RPs	Disabled.
Shortest-path tree threshold rate	0 kb/s.
PIM router query message interval	30 seconds.

Enabling IPV6 PIM Stub Routing

To enable IPV6 PIM stub routing, perform this procedure:

Before you begin

PIM stub routing is disabled in IPv6 by default.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ipv6 multicast pim-passive-enable** **Example:** `Device(config-if)# ipv6 multicast pim-passive-enable`	Enables IPv6 Multicast PIM routing on the switch.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface on which you want to enable PIM stub routing, and enters interface configuration mode. The specified interface must be one of the following: • A routed port—A physical port that has been configured as a Layer 3 port by entering the **no switchport** interface configuration command. You will also need to enable IP PIM sparse mode on the interface, and join the interface as a statically connected member to an MLD static group. • An SVI—A VLAN interface created by using the **interface vlan** *vlan-id* global configuration command. You will also need to enable IP PIM sparse mode on the VLAN, join the VLAN as a statically connected member to an MLD static group, and then enable MLD snooping on the VLAN, the MLD static group, and physical interface. These interfaces must have IPv6 addresses assigned to them.
Step 5	**ipv6 pim** **Example:** `Device(config-if)# ipv6 pim`	Enables the PIM on the interface.
Step 6	**ipv6 pim** {**bsr**} | {**dr-priority** | *value*} | {**hello-interval** | *seconds*} | {**join-prune-interval** | *seconds*} | {**passive**} **Example:** `Device(config-if)# ipv6 pim`	Configures the various PIM stub features on the interface. Enter **bsr** to configure BSR on a PIM switch Enter **dr-priority** to configure the DR priority on a PIM switch.

	Command or Action	Purpose
	bsr|dr-priority|hello-interval|join-prune-interval|passive	Enter **hello-interval** to configure the frequency of PIM hello messages on an interface. Enter **join-prune-interval** to configure periodic join and prune announcement intervals for a specified interface. Enter **passive** to configure the PIM in the passive mode.
Step 7	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Monitoring IPv6 PIM Stub Routing

Table 82: PIM Stub Configuration show Commands

Command	Purpose
show ipv6 pim interface	Displays the PIM stub that is enabled on each interface.
show ipv6 mld groups	Displays the interested clients that have joined the specific multicast source group.
show ipv6 mroute	Verifies that the multicast stream forwards from the source to the interested clients.

Disabling Embedded RP Support in IPv6 PIM

A user might want to disable embedded RP support on an interface if all of the devices in the domain do not support embedded RP.

Note This task disables PIM completely, not just embedded RP support in IPv6 PIM.

To disabling embedded RP support in IPv6 PIM, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no ipv6 pim** [**vrf** *vrf-name*] **rp embedded** **Example:** `Device(config)# no ipv6 pim rp embedded`	Disables embedded RP support in IPv6 PIM.
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 5	**no ipv6 pim** **Example:** `Device(config-if)# no ipv6 pim`	Turns off IPv6 PIM on a specified interface.

Configuring a BSR

The tasks included here are described below.

Configuring a BSR and Verifying BSR Information

To configure and verify BSR Information, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 pim bsr candidate bsr** *ipv6-address*[*hash-mask-length*] [**priority** *priority-value*] **Example:** `Device(config)# ipv6 pim bsr candidate bsr 2001:DB8:3000:3000::42 124 priority 10`	Configures a switch to be a candidate BSR.
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies an interface type and number, and places the switch in interface configuration mode.
Step 5	**ipv6 pim bsr border** **Example:** `Device(config-if)# ipv6 pim bsr border`	Specifies an interface type and number, and places the switch in interface configuration mode.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Enter this command twice to exit interface configuration mode and enter privileged EXEC mode.
Step 7	**show ipv6 pim** bsr {**election** | **rp-cache** | **candidate-rp**} **Example:** `Device(config-if)# show ipv6 pim bsr election`	Displays information related to PIM BSR protocol processing.
Step 8	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Sending PIM RP Advertisements to the BSR

To sending PIM RP advertisements to the BSR, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim bsr candidate rp** *ipv6-address* [**group-list** *access-list-name*] **[priority** *priority-value*] [**interval** seconds] **Example:** `Device(config)# ipv6 pim bsr candidate rp 2001:DB8:3000:3000::42 priority 0`	Sends PIM RP advertisements to the BSR.
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies an interface type and number, and places the switch in interface configuration mode.
Step 5	**ipv6 pim bsr border** **Example:** `Device(config-if)# ipv6 pim bsr border`	Configures a border for all BSMs of any scope on a specified interface.
Step 6	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Configuring BSR for Use Within Scoped Zones

To configure BSR for use within scoped zones, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 pim bsr candidate rp** *ipv6-address* [*hash-mask-length*] [**priority** priority-value] **Example:** `Device(config)# ipv6 pim bsr candidate bsr 2001:DB8:1:1:4`	Configures a switch to be a candidate BSR.
Step 4	**ipv6 pim bsr candidate rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] [**interval** seconds] **Example:** `Device(config)# ipv6 pim bsr candidate rp 2001:DB8:1:1:1 group-list list scope 6`	Configures the candidate RP to send PIM RP advertisements to the BSR.
Step 5	**interface** *type number* **Example:** `Device(config-if)# interface GigabitEthernet 1/1`	Specifies an interface type and number, and places the switch in interface configuration mode.
Step 6	**ipv6 multicast boundary scope** *scope-value* **Example:** `Device(config-if)# ipv6 multicast boundary scope 6`	Configures a multicast boundary on the interface for a specified scope.
Step 7	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Configuring BSR Switches to Announce Scope-to-RP Mappings

IPv6 BSR switches can be statically configured to announce scope-to-RP mappings directly instead of learning them from candidate-RP messages. A user might want to configure a BSR switch to announce scope-to-RP mappings so that an RP that does not support BSR is imported into the BSR. Enabling this feature also allows an RP positioned outside the enterprise's BSR domain to be learned by the known remote RP on the local candidate BSR switch.

To configure BSR switches to announce Scope-to-RP mappings, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim bsr announced rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] **Example:** `Device(config)# ipv6 pim bsr announced rp 2001:DB8:3000:3000::42 priority 0`	Announces scope-to-RP mappings directly from the BSR for the specified candidate RP.
Step 4	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Configuring SSM Mapping

When the SSM mapping feature is enabled, DNS-based SSM mapping is automatically enabled, which means that the switch will look up the source of a multicast MLD version 1 report from a DNS server.

You can use either DNS-based or static SSM mapping, depending on your switch configuration. If you choose to use static SSM mapping, you can configure multiple static SSM mappings. If multiple static SSM mappings are configured, the source addresses of all matching access lists will be used.

Note To use DNS-based SSM mapping, the switch needs to find at least one correctly configured DNS server, to which the switch may be directly attached.

To configuring SSM mapping, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 mld ssm-map enable** **Example:** `Device(config)# ipv6 mld ssm-map enable`	Enables the SSM mapping feature for groups in the configured SSM range.
Step 4	**no ipv6 mld ssm-map query dns** **Example:** `Device(config)# no ipv6 mld ssm-map query dns`	Disables DNS-based SSM mapping.
Step 5	**ipv6 mld ssm-map static** *access-list source-address* **Example:** `Device(config-if)# ipv6 mld ssm-map static SSM_MAP_ACL_2 2001:DB8:1::1`	Configures static SSM mappings.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits global configuration mode, and returns the switch to privileged EXEC mode.
Step 7	**show ipv6 mld ssm-map** [*source-address*] **Example:** `Device(config-if)# show ipv6 mld ssm-map`	Displays SSM mapping information.
Step 8	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Configuring Static Mroutes

Static multicast routes (mroutes) in IPv6 can be implemented as an extension of IPv6 static routes. You can configure your switch to use a static route for unicast routing only, to use a static multicast route for multicast RPF selection only, or to use a static route for both unicast routing and multicast RPF selection.

To configure static mroutes, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route** {*ipv6-prefix / prefix-length ipv6-address* | *interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | *unicast* | *multicast*] [**tag** *tag*] **Example:** `Device(config)# ipv6 route 2001:DB8::/64 6::6 100`	Establishes static IPv6 routes. The example shows a static route used for both unicast routing and multicast RPF selection.
Step 4	**exit** **Example:** `Device# exit`	Exits global configuration mode, and returns the switch to privileged EXEC mode.
Step 5	**show ipv6 mroute** [link-local | [*group-name* | *group-address* [*source-address* | *source-name*]] **[summary**] [**count**] **Example:** `Device# show ipv6 mroute ff07::1`	Displays the contents of the IPv6 multicast routing table.
Step 6	**show ipv6 mroute** [**link-local** | *group-name* | *group-address*] **active** [*kbps*] **Example:** `Device(config-if)# show ipv6 mroute active`	Displays the active multicast streams on the switch.
Step 7	**show ipv6 rpf** [*ipv6-prefix*] **Example:** `Device(config-if)#  show ipv6 rpf 2001::1:1:2`	Checks RPF information for a given unicast host address and prefix.
Step 8	**copy running-config startup-config**	(Optional) Save your entries in the configuration file.

Using MFIB in IPv6 Multicast

Multicast forwarding is automatically enabled when IPv6 multicast routing is enabled.

Verifying MFIB Operation in IPv6 Multicast

To verify MFIB operation in IPv6 multicast

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**show ipv6 mfib** [| **verbose** | *group-address-name* | *ipv6-prefix / prefix-length* | *source-address-name* | **count** | **interface** | **status** | **summary**] **Example:** `Device# show ipv6 mfib`	Displays the forwarding entries and interfaces in the IPv6 MFIB.
Step 3	**show ipv6 mfib** [all | linkscope | group-name | group-address [source-name | source-address]] **count** **Example:** `Device# show ipv6 mfib ff07::1`	Displays the contents of the IPv6 multicast routing table.
Step 4	**show ipv6 mfib interface** **Example:** `Device# show ipv6 mfib interface`	Displays information about IPv6 multicast-enabled interfaces and their forwarding status.
Step 5	**show ipv6 mfib status** **Example:** `Device# show ipv6 mfib status`	Displays general MFIB configuration and operational status.
Step 6	**show ipv6 mfib summary** **Example:** `Device# show ipv6 mfib summary`	Displays summary information about the number of IPv6 MFIB entries and interfaces.
Step 7	**debug ipv6 mfib** [*group-name* | *group-address*] [**adjacency** | **db** | **fs** | **init** | **interface** | **mrib**	Enables debugging output on the IPv6 MFIB.

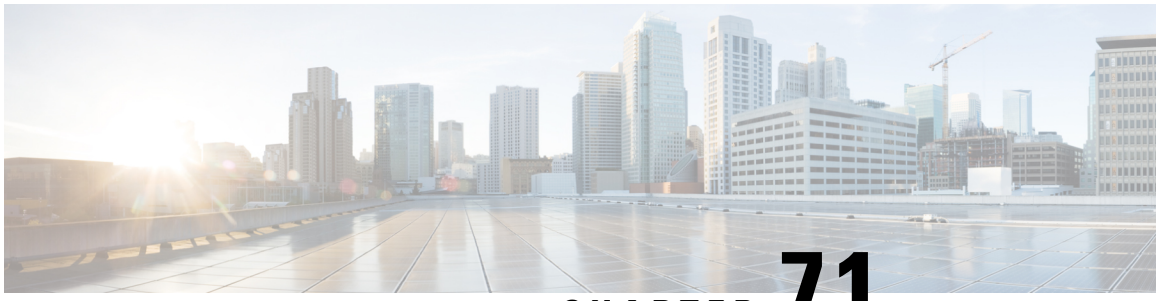
	Command or Action	Purpose
	[**detail**] | **nat** | **pak** | **platform** | **ppr** | **ps** | **signal** | **table**] **Example:** `Device# debug ipv6 mfib FF04::10 pak`	

Resetting MFIB Traffic Counters

To reset MFIB traffic counters, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**clear ipv6 mfib counters** [*group-name* | **group-address** [*source-address* | *source-name*]] **Example:** `Device# clear ipv6 mfib counters FF04::10`	Resets all active MFIB traffic counters.



CHAPTER 71

Configuring MLD Snooping

This module contains details of configuring MLD snooping

Information About Configuring IPv6 MLD Snooping

You can use Multicast Listener Discovery (MLD) snooping to enable efficient distribution of IP Version 6 (IPv6) multicast data to clients and routers in a switched network on the switch. Unless otherwise noted, the term switch refers to a standalone switch.

Understanding MLD Snooping

In IP Version 4 (IPv4), Layer 2 switches can use Internet Group Management Protocol (IGMP) snooping to limit the flooding of multicast traffic by dynamically configuring Layer 2 interfaces so that multicast traffic is forwarded to only those interfaces associated with IP multicast devices. In IPv6, MLD snooping performs a similar function. With MLD snooping, IPv6 multicast data is selectively forwarded to a list of ports that want to receive the data, instead of being flooded to all ports in a VLAN. This list is constructed by snooping IPv6 multicast control packets.

MLD is a protocol used by IPv6 multicast routers to discover the presence of multicast listeners (nodes wishing to receive IPv6 multicast packets) on the links that are directly attached to the routers and to discover which multicast packets are of interest to neighboring nodes. MLD is derived from IGMP; MLD Version 1 (MLDv1) is equivalent to IGMPv2, and MLD Version 2 (MLDv2) is equivalent to IGMPv3. MLD is a subprotocol of Internet Control Message Protocol Version 6 (ICMPv6), and MLD messages are a subset of ICMPv6 messages, identified in IPv6 packets by a preceding Next Header value of 58.

The switch supports two versions of MLD snooping:

- MLDv1 snooping detects MLDv1 control packets and sets up traffic bridging based on IPv6 destination multicast addresses.
- MLDv2 basic snooping (MBSS) uses MLDv2 control packets to set up traffic forwarding based on IPv6 destination multicast addresses.

The switch can snoop on both MLDv1 and MLDv2 protocol packets and bridge IPv6 multicast data based on destination IPv6 multicast addresses.

Note The switch does not support MLDv2 enhanced snooping, which sets up IPv6 source and destination multicast address-based forwarding.

MLD snooping can be enabled or disabled globally or per VLAN. When MLD snooping is enabled, a per-VLAN IPv6 multicast address table is constructed in software and hardware. The switch then performs IPv6 multicast-address based bridging in hardware.

MLD Messages

MLDv1 supports three types of messages:

- Listener Queries are the equivalent of IGMPv2 queries and are either General Queries or Multicast-Address-Specific Queries (MASQs).
- Multicast Listener Reports are the equivalent of IGMPv2 reports
- Multicast Listener Done messages are the equivalent of IGMPv2 leave messages.

MLDv2 supports MLDv2 queries and reports, as well as MLDv1 Report and Done messages.

Message timers and state transitions resulting from messages being sent or received are the same as those of IGMPv2 messages. MLD messages that do not have valid link-local IPv6 source addresses are ignored by MLD routers and switches.

MLD Queries

The switch sends out MLD queries, constructs an IPv6 multicast address database, and generates MLD group-specific and MLD group-and-source-specific queries in response to MLD Done messages. The switch also supports report suppression, report proxying, Immediate-Leave functionality, and static IPv6 multicast group address configuration.

When MLD snooping is disabled, all MLD queries are flooded in the ingress VLAN.

When MLD snooping is enabled, received MLD queries are flooded in the ingress VLAN, and a copy of the query is sent to the CPU for processing. From the received query, MLD snooping builds the IPv6 multicast address database. It detects multicast router ports, maintains timers, sets report response time, learns the querier IP source address for the VLAN, learns the querier port in the VLAN, and maintains multicast-address aging.

When a group exists in the MLD snooping database, the switch responds to a group-specific query by sending an MLDv1 report. When the group is unknown, the group-specific query is flooded to the ingress VLAN.

When a host wants to leave a multicast group, it can send out an MLD Done message (equivalent to IGMP Leave message). When the switch receives an MLDv1 Done message, if Immediate- Leave is not enabled, the switch sends an MASQ to the port from which the message was received to determine if other devices connected to the port should remain in the multicast group.

Multicast Client Aging Robustness

You can configure port membership removal from addresses based on the number of queries. A port is removed from membership to an address only when there are no reports to the address on the port for the configured number of queries. The default number is 2.

Multicast Router Discovery

Like IGMP snooping, MLD snooping performs multicast router discovery, with these characteristics:

- Ports configured by a user never age out.
- Dynamic port learning results from MLDv1 snooping queries and IPv6 PIMv2 packets.
- If there are multiple routers on the same Layer 2 interface, MLD snooping tracks a single multicast router on the port (the router that most recently sent a router control packet).
- Dynamic multicast router port aging is based on a default timer of 5 minutes; the multicast router is deleted from the router port list if no control packet is received on the port for 5 minutes.
- IPv6 multicast router discovery only takes place when MLD snooping is enabled on the switch.
- Received IPv6 multicast router control packets are always flooded to the ingress VLAN, whether or not MLD snooping is enabled on the switch.
- After the discovery of the first IPv6 multicast router port, unknown IPv6 multicast data is forwarded only to the discovered router ports (before that time, all IPv6 multicast data is flooded to the ingress VLAN).

MLD Reports

The processing of MLDv1 join messages is essentially the same as with IGMPv2. When no IPv6 multicast routers are detected in a VLAN, reports are not processed or forwarded from the switch. When IPv6 multicast routers are detected and an MLDv1 report is received, an IPv6 multicast group address is entered in the VLAN MLD database. Then all IPv6 multicast traffic to the group within the VLAN is forwarded using this address. When MLD snooping is disabled, reports are flooded in the ingress VLAN.

When MLD snooping is enabled, MLD report suppression, called listener message suppression, is automatically enabled. With report suppression, the switch forwards the first MLDv1 report received by a group to IPv6 multicast routers; subsequent reports for the group are not sent to the routers. When MLD snooping is disabled, report suppression is disabled, and all MLDv1 reports are flooded to the ingress VLAN.

The switch also supports MLDv1 proxy reporting. When an MLDv1 MASQ is received, the switch responds with MLDv1 reports for the address on which the query arrived if the group exists in the switch on another port and if the port on which the query arrived is not the last member port for the address.

MLD Done Messages and Immediate-Leave

When the Immediate-Leave feature is enabled and a host sends an MLDv1 Done message (equivalent to an IGMP leave message), the port on which the Done message was received is immediately deleted from the group.You enable Immediate-Leave on VLANs and (as with IGMP snooping), you should only use the feature on VLANs where a single host is connected to the port. If the port was the last member of a group, the group is also deleted, and the leave information is forwarded to the detected IPv6 multicast routers.

When Immediate Leave is not enabled in a VLAN (which would be the case when there are multiple clients for a group on the same port) and a Done message is received on a port, an MASQ is generated on that port. The user can control when a port membership is removed for an existing address in terms of the number of MASQs. A port is removed from membership to an address when there are no MLDv1 reports to the address on the port for the configured number of queries.

The number of MASQs generated is configured by using the **ipv6 mld snooping last-listener-query count** global configuration command. The default number is 2.

The MASQ is sent to the IPv6 multicast address for which the Done message was sent. If there are no reports sent to the IPv6 multicast address specified in the MASQ during the switch maximum response time, the port on which the MASQ was sent is deleted from the IPv6 multicast address database. The maximum response time is the time configured by using the **ipv6 mld snooping last-listener-query-interval** global configuration command. If the deleted port is the last member of the multicast address, the multicast address is also deleted, and the switch sends the address leave information to all detected multicast routers.

Topology Change Notification Processing

When topology change notification (TCN) solicitation is enabled by using the **ipv6 mld snooping tcn query solicit** global configuration command, MLDv1 snooping sets the VLAN to flood all IPv6 multicast traffic with a configured number of MLDv1 queries before it begins sending multicast data only to selected ports. You set this value by using the **ipv6 mld snooping tcn flood query count** global configuration command. The default is to send two queries. The switch also generates MLDv1 global Done messages with valid link-local IPv6 source addresses when the switch becomes the STP root in the VLAN or when it is configured by the user. This is same as done in IGMP snooping.

How to Configure IPv6 MLD Snooping

Default MLD Snooping Configuration

Table 83: Default MLD Snooping Configuration

Feature	Default Setting
MLD snooping (Global)	Disabled.
MLD snooping (per VLAN)	Enabled. MLD snooping must be globally enabled for VLAN MLD snooping to take place.
IPv6 Multicast addresses	None configured.
IPv6 Multicast router ports	None configured.
MLD snooping Immediate Leave	Disabled.
MLD snooping robustness variable	Global: 2; Per VLAN: 0. **Note** The VLAN value overrides the global setting. When the VLAN value is 0, the VLAN uses the global count.
Last listener query count	Global: 2; Per VLAN: 0. **Note** The VLAN value overrides the global setting. When the VLAN value is 0, the VLAN uses the global count.

Feature	Default Setting
Last listener query interval	Global: 1000 (1 second); VLAN: 0. **Note** The VLAN value overrides the global setting. When the VLAN value is 0, the VLAN uses the global interval.
TCN query solicit	Disabled.
TCN query count	2.
MLD listener suppression	Disabled.

MLD Snooping Configuration Guidelines

When configuring MLD snooping, consider these guidelines:

- You can configure MLD snooping characteristics at any time, but you must globally enable MLD snooping by using the **ipv6 mld snooping** global configuration command for the configuration to take effect.
- MLD snooping and IGMP snooping act independently of each other. You can enable both features at the same time on the switch.
- The maximum number of multicast entries allowed for the switch is 1024.

Enabling or Disabling MLD Snooping on the Switch

By default, IPv6 MLD snooping is globally disabled on the switch and enabled on all VLANs. When MLD snooping is globally disabled, it is also disabled on all VLANs. When you globally enable MLD snooping, the VLAN configuration overrides the global configuration. That is, MLD snooping is enabled only on VLAN interfaces in the default state (enabled).

You can enable and disable MLD snooping on a per-VLAN basis or for a range of VLANs, but if you globally disable MLD snooping, it is disabled in all VLANs. If global snooping is enabled, you can enable or disable VLAN snooping.

To globally enable MLD snooping on the switch, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ipv6 mld snooping** **Example:** `Device(config)# ipv6 mld snooping`	Enables MLD snooping on the switch.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**copy running-config startup-config** **Example:** `Device(config)# copy running-config startup-config`	(Optional) Save your entries in the configuration file.
Step 6	**reload** **Example:** `Device(config)# reload`	Reload the operating system.

Enabling or Disabling MLD Snooping on a VLAN

To enable MLD snooping on a VLAN, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 mld snooping** **Example:** `Device(config)# ipv6 mld snooping`	Enables MLD snooping on the switch.
Step 4	**ipv6 mld snooping vlan** *vlan-id* **Example:** `Device(config)# ipv6 mld snooping vlan 1`	Enables MLD snooping on the VLAN. The VLAN ID range is 1 to 1001 and 1006 to 4094. **Note** MLD snooping must be globally enabled for VLAN snooping to be enabled.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring a Static Multicast Group

Hosts or Layer 2 ports normally join multicast groups dynamically, but you can also statically configure an IPv6 multicast address and member ports for a VLAN.

To add a Layer 2 port as a member of a multicast group, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mld snooping vlan** *vlan-id* **static** *ipv6_multicast_address* **interface** *interface-id* **Example:** `Device(config)# ipv6 mld snooping vlan 1 static 3333.0000.1111 interface`	Configures a multicast group with a Layer 2 port as a member of a multicast group: • *vlan-id* is the multicast group VLAN ID. The VLAN ID range is 1 to 1001 and 1006 to 4094.

	Command or Action	Purpose
	gigabitethernet 1/1	• *ipv6_multicast_address* is the 128-bit group IPv6 address. The address must be in the form specified in RFC 2373. • *interface-id* is the member port. It can be a physical interface or a port channel (1 to 48).
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	Use one of the following: • **show ipv6 mld snooping address** • **show ipv6 mld snooping address vlan** *vlan-id* **Example:** `Device# show ipv6 mld snooping address` or `Device# show ipv6 mld snooping vlan 1`	Verifies the static member port and the IPv6 address.

Configuring a Multicast Router Port

Note Static connections to multicast routers are supported only on switch ports.

To add a multicast router port to a VLAN, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ipv6 mld snooping vlan** *vlan-id* **mrouter interface** *interface-id* **Example:** `Device(config)# ipv6 mld snooping vlan 1 mrouter interface gigabitethernet 1/2`	Specifies the multicast router VLAN ID, and specify the interface to the multicast router. • The VLAN ID range is 1 to 1001 and 1006 to 4094. • The interface can be a physical interface or a port channel. The port-channel range is 1 to 48.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show ipv6 mld snooping mrouter** [**vlan** *vlan-id*] **Example:** `Device# show ipv6 mld snooping mrouter vlan 1`	Verifies that IPv6 MLD snooping is enabled on the VLAN interface.

Enabling MLD Immediate Leave

To enable MLDv1 immediate leave, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mld snooping vlan** *vlan-id* **immediate-leave** **Example:** `Device(config)# ipv6 mld snooping vlan 1 immediate-leave`	Enables MLD Immediate Leave on the VLAN interface.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show ipv6 mld snooping vlan** *vlan-id* **Example:** `Device# show ipv6 mld snooping vlan 1`	Verifies that Immediate Leave is enabled on the VLAN interface.

Configuring MLD Snooping Queries

To configure MLD snooping query characteristics for the switch or for a VLAN, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	[**no**] **ipv6 mld snooping robustness-variable** *value* **Example:** `Device(config)# ipv6 mld snooping robustness-variable 3`	(Optional) Sets the number of queries that are sent before switch will deletes a listener (port) that does not respond to a general query. The range is 1 to 3; the default is 2. Use the no form of this command to disable this feature.
Step 4	**ipv6 mld snooping vlan** *vlan-id* **robustness-variable** *value* **Example:** `Device(config)# ipv6 mld snooping vlan 1 robustness-variable 3`	(Optional) Sets the robustness variable on a VLAN basis, which determines the number of general queries that MLD snooping sends before aging out a multicast address when there is no MLD report response. The range is 1 to 3; the default is 0. When set to 0, the number that is used is the global robustness variable value.
Step 5	[**no**] **ipv6 mld snooping last-listener-query-count** *count*	(Optional) Sets the number of MASQs that the switch sends before aging out an MLD client.

	Command or Action	Purpose
	Example: `Device(config)# ipv6 mld snooping last-listener-query-count 7`	The range is 1 to 7; the default is 2. The queries are sent 1 second apart. Use the no form of this command to disable this feature.
Step 6	**ipv6 mld snooping vlan** *vlan-id* **last-listener-query-count** *count* **Example:** `Device(config)# ipv6 mld snooping vlan 1 last-listener-query-count 7`	(Optional) Sets the last-listener query count on a VLAN basis. This value overrides the value that is configured globally. The range is 1 to 7; the default is 0. When set to 0, the global count value is used. Queries are sent 1 second apart.
Step 7	[**no**] **ipv6 mld snooping last-listener-query-interval** *interval* **Example:** `Device(config)# ipv6 mld snooping last-listener-query-interval 2000`	(Optional) Sets the maximum response time that the switch waits after sending out a MASQ before deleting a port from the multicast group. The range is 100 to 32,768 thousands of a second. The default is 1000 (1 second). Use the no form of this command to disable this feature.
Step 8	**ipv6 mld snooping vlan** *vlan-id* **last-listener-query-interval** *interval* **Example:** `Device(config)# ipv6 mld snooping vlan 1 last-listener-query-interval 2000`	(Optional) Sets the last-listener query interval on a VLAN basis. This value overrides the value that is configured globally. The range is 0 to 32,768 thousands of a second. The default is 0. When set to 0, the global last-listener query interval is used.
Step 9	[**no**] **ipv6 mld snooping tcn query solicit** **Example:** `Device(config)# ipv6 mld snooping tcn query solicit`	(Optional) Enables topology change notification (TCN) solicitation, which means that VLANs flood all IPv6 multicast traffic for the configured number of queries before sending multicast data to only those ports requesting to receive it. The default is for TCN to be disabled. Use the no form of this command to disable this feature.
Step 10	[**no**] **ipv6 mld snooping tcn flood query count** *count* **Example:** `Device(config)# ipv6 mld snooping tcn flood query count 5`	(Optional) When TCN is enabled, specifies the number of TCN queries to be sent. The range is from 1 to 10; the default is 2. Use the no form of this command to disable this feature.
Step 11	[**no**] **ipv6 mld snooping querier** **Example:**	(Optional) Enables MLD snooping queries. Use the no form of this command to disable MLD snooping queries.

	Command or Action	Purpose
	`Device(config)# ipv6 mld snooping querier`	
Step 12	[**no**] **ipv6 mld snooping vlan** *vlan_id* **querier** **Example:** `Device(config)# ipv6 mld snooping vlan 1 querier`	(Optional) Enables MLD snooping queries in a VLAN. Use the no form of this command to disable MLD snooping queries in a VLAN.
Step 13	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 14	**show ipv6 mld snooping querier** [**vlan** *vlan-id*] **Example:** `Device# show ipv6 mld snooping querier vlan 1`	(Optional) Verifies that the MLD snooping querier information for the switch or for the VLAN.

Disabling MLD Listener Message Suppression

MLD snooping listener message suppression is enabled by default. When it is enabled, the switch forwards only one MLD report per multicast router query. When message suppression is disabled, multiple MLD reports could be forwarded to the multicast routers.

To disable MLD listener message suppression, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enter global configuration mode.
Step 3	**no ipv6 mld snooping listener-message-suppression** **Example:**	Disable MLD message suppression.

	Command or Action	Purpose
	`Device(config)# no ipv6 mld snooping listener-message-suppression`	
Step 4	**end** **Example:** `Device(config)# end`	Return to privileged EXEC mode.
Step 5	**show ipv6 mld snooping** **Example:** `Device# show ipv6 mld snooping`	Verify that IPv6 MLD snooping report suppression is disabled.

Monitoring MLD Snooping Information

You can display MLD snooping information for dynamically learned and statically configured router ports and VLAN interfaces. You can also display IPv6 group address multicast entries for a VLAN configured for MLD snooping.

Table 84: Commands for Displaying MLD Snooping Information

Command	Purpose
show ipv6 mld snooping [**vlan** *vlan-id*]	Displays the MLD snooping configuration information for all VLANs on the switch or for a specified VLAN. (Optional) Enter **vlan** *vlan-id* to display information for a single VLAN. The VLAN ID range is 1 to 1001 and 1006 to 4094.
show ipv6 mld snooping mrouter [**vlan** *vlan-id*]	Displays information on dynamically learned and manually configured multicast router interfaces. When you enable MLD snooping, the switch automatically learns the interface to which a multicast router is connected. These are dynamically learned interfaces. (Optional) Enters **vlan** *vlan-id* to display information for a single VLAN. The VLAN ID range is 1 to 1001 and 1006 to 4094.
show ipv6 mld snooping querier [**vlan** *vlan-id*]	Displays information about the IPv6 address and incoming port for the most-recently received MLD query messages in the VLAN. (Optional) Enters **vlan** *vlan-id* to display information for a single VLAN.The VLAN ID range is 1 to 1001 and 1006 to 4094.

Command	Purpose
show ipv6 mld snooping address [**vlan** *vlan-id*] [**count** | **dynamic** | **user**]	Displays all IPv6 multicast address information or specific IPv6 multicast address information for the switch or a VLAN. • Enters **count** to show the group count on the switch or in a VLAN. • Enters **dynamic** to display MLD snooping learned group information for the switch or for a VLAN. • Enters **user** to display MLD snooping user-configured group information for the switch or for a VLAN.
show ipv6 mld snooping address vlan *vlan-id* [*ipv6-multicast-address*]	Displays MLD snooping for the specified VLAN and IPv6 multicast address.

Configuration Examples for Configuring MLD Snooping

Configuring a Static Multicast Group: Example

This example shows how to statically configure an IPv6 multicast group:

```
Device# configure terminal
Device(config)# ipv6 mld snooping vlan 2 static 3333.0000.1111 interface gigabitethernet
1/1
Device(config)# end
```

Configuring a Multicast Router Port: Example

This example shows how to add a multicast router port to VLAN 200:

```
Device# configure terminal
Device(config)# ipv6 mld snooping vlan 200 mrouter interface gigabitethernet 1/2
Device(config)# exit
```

Enabling MLD Immediate Leave: Example

This example shows how to enable MLD Immediate Leave on VLAN 130:

```
Device# configure terminal
Device(config)# ipv6 mld snooping vlan 130 immediate-leave
Device(config)# exit
```

Configuring MLD Snooping Queries: Example

This example shows how to set the MLD snooping global robustness variable to 3:

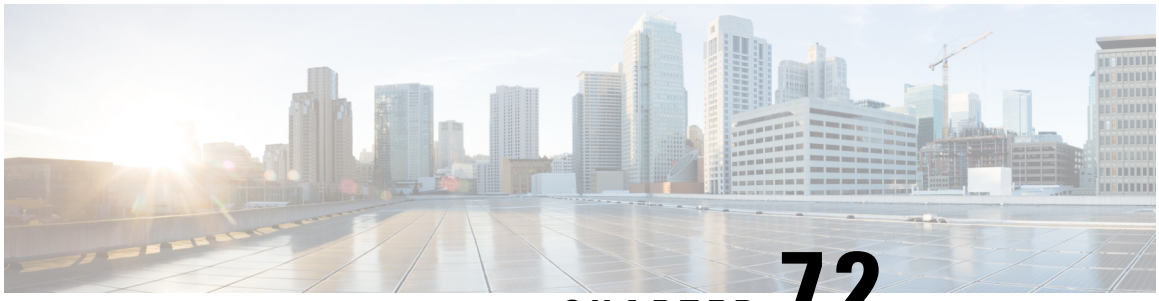
```
Device# configure terminal
Device(config)# ipv6 mld snooping robustness-variable 3
Device(config)# exit
```

This example shows how to set the MLD snooping last-listener query count for a VLAN to 3:

```
Device# configure terminal
Device(config)# ipv6 mld snooping vlan 200 last-listener-query-count 3
Device(config)# exit
```

This example shows how to set the MLD snooping last-listener query interval (maximum response time) to 2000 (2 seconds):

```
Device# configure terminal
Device(config)# ipv6 mld snooping last-listener-query-interval 2000
Device(config)# exit
```



CHAPTER 72

IP Multicast Optimization: Optimizing PIM Sparse Mode in a Large IP Multicast Deployment

Prerequisites for Optimizing PIM Sparse Mode in a Large IP Multicast Deployment

- You must have PIM sparse mode running in your network.
- If you plan to use a group list to control to which groups the shortest-path tree (SPT) threshold applies, you must have configured your access list before performing the task.

Information About Optimizing PIM Sparse Mode in a Large IP Multicast Deployment

PIM Registering Process

IP multicast sources do not use a signaling mechanism to announce their presence. Sources just send their data into the attached network, as opposed to receivers that use Internet Group Management Protocol (IGMP) to announce their presence. If a source sends traffic to a multicast group configured in PIM sparse mode (PIM-SM), the Designated Router (DR) leading toward the source must inform the rendezvous point (RP) about the presence of this source. If the RP has downstream receivers that want to receive the multicast traffic (natively) from this source and has not joined the shortest path leading toward the source, then the DR must send the traffic from the source to the RP. The PIM registering process, which is individually run for each (S, G) entry, accomplishes these tasks between the DR and RP.

The registering process begins when a DR creates a new (S, G) state. The DR encapsulates all the data packets that match the (S, G) state into PIM register messages and unicasts those register messages to the RP.

If an RP has downstream receivers that want to receive register messages from a new source, the RP can either continue to receive the register messages through the DR or join the shortest path leading toward the source. By default, the RP will join the shortest path, because delivery of native multicast traffic provides the highest throughput. Upon receipt of the first packet that arrives natively through the shortest path, the RP will send a register-stop message back to the DR. When the DR receives this register-stop message, it will stop sending register messages to the RP.

If an RP has no downstream receivers that want to receive register messages from a new source, the RP will not join the shortest path. Instead, the RP will immediately send a register-stop message back to the DR. When the DR receives this register-stop message, it will stop sending register messages to the RP.

Once a routing entry is established for a source, a periodic reregistering takes place between the DR and RP. One minute before the multicast routing table state times out, the DR will send one dataless register message to the RP each second that the source is active until the DR receives a register-stop message from the RP. This action restarts the timeout time of the multicast routing table entry, typically resulting in one reregistering exchange every 2 minutes. Reregistering is necessary to maintain state, to recover from lost state, and to keep track of sources on the RP. It will take place independently of the RP joining the shortest path.

PIM Version 1 Compatibility

If an RP is running PIM Version 1, it will not understand dataless register messages. In this case, the DR will not send dataless register messages to the RP. Instead, approximately every 3 minutes after receipt of a register-stop message from the RP, the DR encapsulates the incoming data packets from the source into register messages and sends them to the RP. The DR continues to send register messages until it receives another register-stop message from the RP. The same behavior occurs if the DR is running PIM Version 1.

When a DR running PIM Version 1 encapsulates data packets into register messages for a specific (S, G) entry, the entry is process-switched, not fast-switched or hardware-switched. On platforms that support these faster paths, the PIM registering process for an RP or DR running PIM Version 1 may lead to periodic out-of-order packet delivery. For this reason, we recommend upgrading your network from PIM Version 1 to PIM Version 2.

PIM Designated Router

Devices configured for IP multicast send PIM hello messages to determine which device will be the designated router (DR) for each LAN segment (subnet). The hello messages contain the device's IP address, and the device with the highest IP address becomes the DR.

The DR sends Internet Group Management Protocol (IGMP) host query messages to all hosts on the directly connected LAN. When operating in sparse mode, the DR sends source registration messages to the rendezvous point (RP).

By default, multicast devices send PIM router query messages every 30 seconds. By enabling a device to send PIM hello messages more often, the device can discover unresponsive neighbors more quickly. As a result, the device can implement failover or recovery procedures more efficiently. It is appropriate to make this change only on redundant devices on the edge of the network.

PIM Sparse-Mode Register Messages

Dataless register messages are sent at a rate of one message per second. Continuous high rates of register messages might occur if a DR is registering bursty sources (sources with high data rates) and if the RP is not running PIM Version 2.

By default, PIM sparse-mode register messages are sent without limiting their rate. Limiting the rate of register messages will limit the load on the DR and RP, at the expense of dropping those register messages that exceed the set limit. Receivers may experience data packet loss within the first second in which packets are sent from bursty sources.

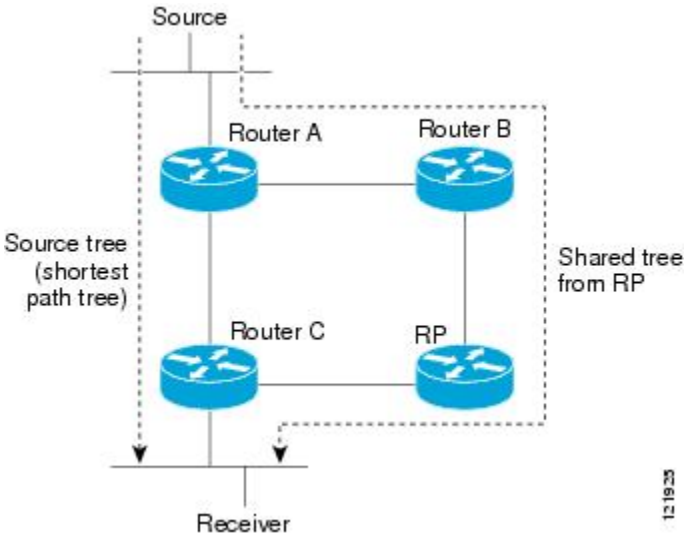
Preventing Use of Shortest-Path Tree to Reduce Memory Requirement

Understanding PIM shared tree and source tree will help you understand how preventing the use of the shortest-path tree can reduce memory requirements.

PIM Shared Tree and Source Tree - Shortest-Path Tree

By default, members of a multicast group receive data from senders to the group across a single data distribution tree rooted at the rendezvous point (RP). This type of distribution tree is called shared tree, as shown in the figure. Data from senders is delivered to the RP for distribution to group members joined to the shared tree.

Figure 80: Shared Tree versus Source Tree (Shortest-Path Tree)



If the data rate warrants, leaf routers on the shared tree may initiate a switch to the data distribution tree rooted at the source. This type of distribution tree is called a shortest-path tree (SPT) or source tree. By default, the software switches to a source tree upon receiving the first data packet from a source.

The following process describes the move from shared tree to source tree in more detail:

1. Receiver joins a group; leaf Router C sends a Join message toward the RP.
2. The RP puts the link to Router C in its outgoing interface list.
3. Source sends data; Router A encapsulates data in a register message and sends it to the RP.

4. The RP forwards data down the shared tree to Router C and sends a Join message toward the source. At this point, data may arrive twice at Router C, once encapsulated and once natively.
5. When data arrives natively (through multicast) at the RP, the RP sends a register-stop message to Router A.
6. By default, reception of the first data packet prompts Router C to send a Join message toward the source.
7. When Router C receives data on (S, G), it sends a Prune message for the source up the shared tree.
8. The RP deletes the link to Router C from the outgoing interface of (S, G). The RP triggers a Prune message toward the source.

Join and Prune messages are sent for sources and RPs. They are sent hop-by-hop and are processed by each PIM router along the path to the source or RP. Register and register-stop messages are not sent hop-by-hop. They are sent by the designated router that is directly connected to a source and are received by the RP for the group.

Multiple sources sending to groups use the shared tree.

Benefit of Preventing or Delaying the Use of the Shortest-Path Tree

The switch from shared to source tree happens upon the arrival of the first data packet at the last hop device (Router C in PIM Shared Tree and Source Tree - Shortest-Path Tree, on page 1025). This switch occurs because the **ip pim spt-threshold** command controls that timing, and its default setting is 0 kbps.

The shortest-path tree requires more memory than the shared tree, but reduces delay. You might want to prevent or delay its use to reduce memory requirements. Instead of allowing the leaf device to move to the shortest-path tree immediately, you can prevent use of the SPT or specify that the traffic must first reach a threshold.

You can configure when a PIM leaf device should join the shortest-path tree for a specified group. If a source sends at a rate greater than or equal to the specified *kbps* rate, the device triggers a PIM Join message toward the source to construct a source tree (shortest-path tree). If the **infinity** keyword is specified, all sources for the specified group use the shared tree, never switching to the source tree.

How to Optimize PIM Sparse Mode in a Large IP Multicast Deployment

Optimizing PIM Sparse Mode in a Large Deployment

Consider performing this task if your deployment of IP multicast is large.

Steps 3, 5, and 6 in this task are independent of each other and are therefore considered optional. Any one of these steps will help optimize PIM sparse mode. If you are going to perform Step 5 or 6, you must perform Step 4. Step 6 applies only to a designated router; changing the PIM query interval is only appropriate on redundant routers on the edge of the PIM domain.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip pim register-rate-limit** *rate* **Example:** `Router(config)# ip pim register-rate-limit 10`	(Optional) Sets a limit on the maximum number of PIM sparse mode register messages sent per second for each (S, G) routing entry. • Use this command to limit the number of register messages that the designated router (DR) will allow for each (S, G) entry. • By default, there is no maximum rate set. • Configuring this command will limit the load on the DR and RP at the expense of dropping those register messages that exceed the set limit. • Receivers may experience data packet loss within the first second in which register messages are sent from bursty sources.
Step 4	**ip pim spt-threshold** {*kbps*| **infinity**}[**group-list** *access-list*] **Example:** `Router(config)# ip pim spt-threshold infinity group-list 5`	(Optional) Specifies the threshold that must be reached before moving to the shortest-path tree. • The default value is **0**, which causes the router to join the SPT immediately upon the first data packet it receives. • Specifying the **infinity** keyword causes the router never to move to the shortest-path tree; it remains on the shared tree. This keyword applies to a multicast environment of "many-to-many" communication. • The group list is a standard access list that controls which groups the SPT threshold applies to. If a value of 0 is specified or the group list is not used, the threshold applies to all groups.

	Command or Action	Purpose
		• In the example, group-list 5 is already configured to permit the multicast groups 239.254.2.0 and 239.254.3.0: access-list 5 permit 239.254.2.0 0.0.0.255 access-list 5 permit 239.254.3.0 0.0.0.255
Step 5	**interface** *type number* **Example:** `Router(config)# interface Gigabitethernet 1/1`	Configures an interface. • If you do not want to change the default values of the PIM SPT threshold or the PIM query interval, do not perform this step; you are done with this task.
Step 6	**ip pim query-interval** *period* [**msec**] **Example:** `Router(config-if)# ip pim query-interval 1`	(Optional) Configures the frequency at which multicast routers send PIM router query messages. • Perform this step only on redundant routers on the edge of a PIM domain. • The default query interval is 30 seconds. • The *period* argument is in seconds unless the **msec** keyword is specified. • Set the query interval to a smaller number of seconds for faster convergence, but keep in mind the trade-off between faster convergence and higher CPU and bandwidth usage.

Configuration Examples for Optimizing PIM Sparse Mode in a Large Multicast Deployment

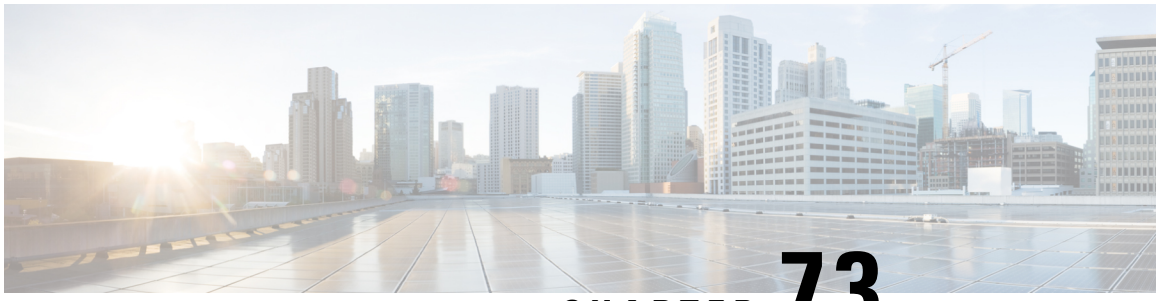
Optimizing PIM Sparse Mode in a Large IP Multicast Deployment Example

The following example shows how to:

- Set the query interval to 1 second for faster convergence.
- Configure the router to never move to the SPT but to remain on the shared tree.
- Set a limit of 10 PIM sparse mode register messages sent per second for each (S, G) routing entry.

```
interface Gigabitethernet 1/1
 ip pim query-interval 1
.
.
```

```
.
!
ip pim spt-threshold infinity
ip pim register-rate-limit 10
!
```



CHAPTER 73

IP Multicast Optimization: IP Multicast Load Splitting across Equal-Cost Paths

Prerequisites for IP Multicast Load Splitting across Equal-Cost Paths

IP multicast is enabled on the device using the tasks described in the "Configuring Basic IP Multicast" module of the *IP Multicast Routing Configuration Guide*.

Information About IP Multicast Load Splitting across Equal-Cost Paths

Load Splitting Versus Load Balancing

Load splitting and load balancing are not the same. Load splitting provides a means to randomly distribute (*, G) and (S, G) traffic streams across multiple equal-cost reverse path forwarding (RPF) paths, which does not necessarily result in a balanced IP multicast traffic load on those equal-cost RPF paths. By randomly distributing (*, G) and (S, G) traffic streams, the methods used for load splitting IP multicast traffic attempt to distribute an equal amount of traffic flows on each of the available RPF paths not by counting the flows, but, rather, by making a pseudorandom decision. These methods are collectively referred to as equal-cost multipath (ECMP) multicast load splitting methods and result in better load-sharing in networks where there are many traffic streams that utilize approximately the same amount of bandwidth.

If there are just a few (S, G) or (*, G) states flowing across a set of equal-cost links, the chance that they are well balanced is quite low. To overcome this limitation, precalculated source addresses--for (S, G) states or rendezvous point (RP) addresses for (*, G) states, can be used to achieve a reasonable form of load balancing. This limitation applies equally to the per-flow load splitting in Cisco Express Forwarding (CEF) or with

EtherChannels: As long as there are only a few flows, those methods of load splitting will not result in good load distribution without some form of manual engineering.

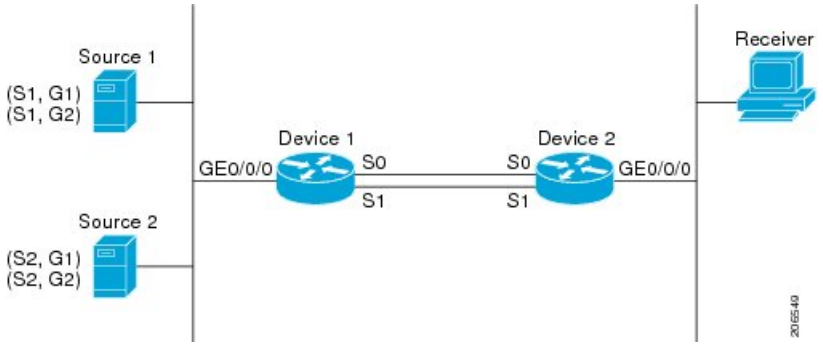
Default Behavior for IP Multicast When Multiple Equal-Cost Paths Exist

By default, for Protocol Independent Multicast sparse mode (PIM-SM), Source Specific Multicast (PIM-SSM), bidirectional PIM (bidir-PIM), groups, if multiple equal-cost paths are available, Reverse Path Forwarding (RPF) for IPv4 multicast traffic is based on the PIM neighbor with the highest IP address. This method is referred to as the highest PIM neighbor behavior. This behavior is in accordance with RFC 2362 for PIM-SM, but also applies to PIM-SSM, and bidir-PIM.

The figure illustrates a sample topology that is used in this section to explain the default behavior for IP multicast when multiple equal-cost paths exist.

Note Although the following illustration and example uses routers in the configuration, any device (router or controller) can be used.

Figure 81: Default Behavior for IP Multicast When Multiple Equal-Cost Paths Exist



In the figure, two sources, S1 and S2, are sending traffic to IPv4 multicast groups, G1 and G2. Either PIM-SM, PIM-SSM can be used in this topology. If PIM-SM is used, assume that the default of 0 for the **ip pim spt-threshold** command is being used on Device 2, that an Interior Gateway Protocol (IGP) is being run, and that the output of the **show ip route** command for S1 and for S2 (when entered on Device 2) displays serial interface 0 and serial interface 1 on Device 1 as equal-cost next-hop PIM neighbors of Device 2.

Without further configuration, IPv4 multicast traffic in the topology illustrated in the figure would always flow across one serial interface (either serial interface 0 or serial interface 1), depending on which interface has the higher IP address. For example, suppose that the IP addresses configured on serial interface 0 and serial interface 1 on Device 1 are 10.1.1.1 and 10.1.2.1, respectively. Given that scenario, in the case of PIM-SM and PIM-SSM, Device 2 would always send PIM join messages towards 10.1.2.1 and would always receive IPv4 multicast traffic on serial interface 1 for all sources and groups shown in the figure.

IPv4 RPF lookups are performed by intermediate multicast device to determine the RPF interface and RPF neighbor for IPv4 (*,G) and (S, G) multicast routes (trees). An RPF lookup consists of RPF route-selection and route-path-selection. RPF route-selection operates solely on the IP unicast address to identify the root of the multicast tree. For (*, G) routes (PIM-SM and Bidir-PIM), the root of the multicast tree is the RP address for the group G; for (S, G) trees (PIM-SM, PIM-SSM, the root of the multicast tree is the source S. RPF route-selection finds the best route towards the RP or source in the routing information base (RIB), and, if configured (or available), the Distance Vector Multicast Routing Protocol (DVMRP) routing table or configured

static mroutes. If the resulting route has only one available path, then the RPF lookup is complete, and the next-hop device and interface of the route become the RPF neighbor and RPF interface of this multicast tree. If the route has more than one path available, then route-path-selection is used to determine which path to choose.

For IP multicast, the following route-path-selection methods are available:

Note All methods but the default method of route-path-selection available in IP multicast enable some form of ECMP multicast load splitting.

- Highest PIM neighbor--This is the default method; thus, no configuration is required. If multiple equal-cost paths are available, RPF for IPv4 multicast traffic is based on the PIM neighbor with the highest IP address; as a result, without configuration, ECMP multicast load splitting is disabled by default.
- ECMP multicast load splitting method based on source address--You can configure ECMP multicast load splitting using the **ip multicast multipath** command. Entering this form of the **ip multicast multipath** command enables ECMP multicast load splitting based on source address using the S-hash algorithm. For more information, see the *ECMP Multicast Load Splitting Based on Source Address Using the S-Hash Algorithm* section.
- ECMP multicast load splitting method based on source and group address--You can configure ECMP multicast load splitting using the **ip multicast multipath** command with the **s-g-hash** and **basic** keywords. Entering this form of the **ip multicast multipath** command enables ECMP multicast load splitting based on source and group address using the basic S-G-hash algorithm. For more information, see the *ECMP Multicast Load Splitting Based on Source and Group Address Using the Basic S-G-Hash Algorithm* section.
- ECMP multicast load splitting method based on source, group, and next-hop address--You can configure ECMP multicast load splitting using the **ip multicast multipath** command with the **s-g-hash** and **next-hop-based** keywords. Entering this form of the command enables ECMP multicast load splitting based on source, group, and next-hop address using the next-hop-based S-G-hash algorithm. For more information, see the *ECMP Multicast Load Splitting Based on Source Group and Next-Hop Address* section.

The default behavior (the highest PIM neighbor behavior) does not result in any form of ECMP load-splitting in IP multicast, but instead selects the PIM neighbor that has the highest IP address among the next-hop PIM neighbors for the available paths. A next hop is considered to be a PIM neighbor when it displays in the output of the **show ip pim neighbor** command, which is the case when PIM hello messages have been received from it and have not timed out. If none of the available next hops are PIM neighbors, then simply the next hop with the highest IP address is chosen.

Methods to Load Split IP Multicast Traffic

In general, the following methods are available to load split IP multicast traffic:

- You can enable ECMP multicast load splitting based on source address, based on source and group address, or based on source, group, and next-hop address. After the equal-cost paths are recognized, ECMP multicast load splitting operates on a per (S, G) basis, rather than a per packet basis as in unicast traffic.

Overview of ECMP Multicast Load Splitting

By default, ECMP multicast load splitting of IPv4 multicast traffic is disabled. ECMP multicast load splitting can be enabled using the **ip multicast multipath** command.

ECMP Multicast Load Splitting Based on Source Address Using the S-Hash Algorithm

ECMP multicast load splitting traffic based on source address uses the S-hash algorithm, enabling the RPF interface for each (*, G) or (S, G) state to be selected among the available equal-cost paths, depending on the RPF address to which the state resolves. For an (S, G) state, the RPF address is the source address of the state; for a (*, G) state, the RPF address is the address of the RP associated with the group address of the state.

When ECMP multicast load splitting based on source address is configured, multicast traffic for different states can be received across more than just one of the equal-cost interfaces. The method applied by IPv4 multicast is quite similar in principle to the default per-flow load splitting in IPv4 CEF or the load splitting used with Fast and Gigabit EtherChannels. This method of ECMP multicast load splitting, however, is subject to polarization.

ECMP Multicast Load Splitting Based on Source and Group Address Using the Basic S-G-Hash Algorithm

ECMP multicast load splitting based on source and group address uses a simple hash, referred to as the basic S-G-hash algorithm, which is based on source and group address. The basic S-G-hash algorithm is predictable because no randomization is used in coming up with the hash value. The S-G-hash mechanism, however, is subject to polarization because for a given source and group, the same hash is always picked irrespective of the device this hash is being calculated on.

Note The basic S-G-hash algorithm ignores bidir-PIM groups.

Predictability As a By-Product of Using the S-Hash and Basic S-G-Hash Algorithms

The method used by ECMP multicast load splitting in IPv4 multicast allows for consistent load splitting in a network where the same number of equal-cost paths are present in multiple places in a topology. If an RP address or source addresses are calculated once to have flows split across N paths, then they will be split across those N paths in the same way in all places in the topology. Consistent load splitting allows for predictability, which, in turn, enables load splitting of IPv4 multicast traffic to be manually engineered.

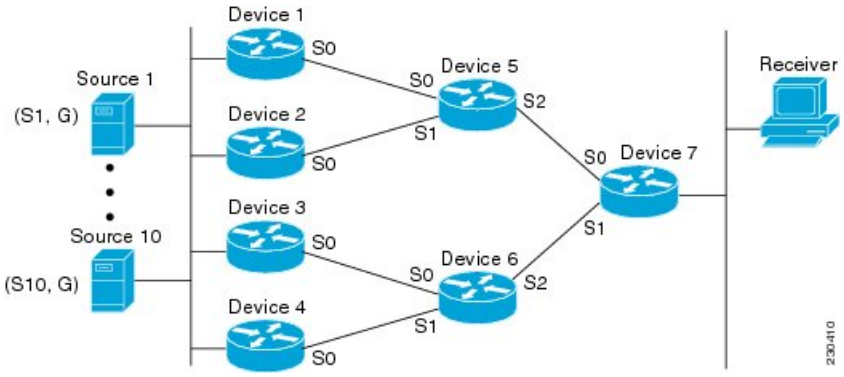
Polarization As a By-Product of Using the S-Hash and Basic S-G-Hash Algorithms

The hash mechanism used in IPv4 multicast to load split multicast traffic by source address or by source and group address is subject to a problem usually referred to as polarization. A by-product of ECMP multicast load splitting based on source address or on source and group address, polarization is a problem that prevents routers in some topologies from effectively utilizing all available paths for load splitting.

The figure illustrates a sample topology that is used in this section to explain the problem of polarization when configuring ECMP multicast load splitting based on source address or on source and group address.

Note Although the following illustration and example uses routers in the configuration, any device (router or switch) can be used.

Figure 82: Polarization Topology



In the topology illustrated in the figure, notice that Router 7 has two equal-cost paths towards the sources, S1 to S10, through Router 5 and Router 6. For this topology, suppose that ECMP multicast load splitting is enabled with the **ip multicast multipath** command on all routers in the topology. In that scenario, Router 7 would apply equal-cost load splitting to the 10 (S, G) states. The problem of polarization in this scenario would affect Router 7 because that router would end up choosing serial interface 0 on Router 5 for sources S1 to S5 and serial interface 1 on Router 6 for sources S6 to S10. The problem of polarization, furthermore, would also affect Router 5 and Router 6 in this topology. Router 5 has two equal-cost paths for S1 to S5 through serial interface 0 on Router 1 and serial interface 1 on Router 2. Because Router 5 would apply the same hash algorithm to select which of the two paths to use, it would end up using just one of these two upstream paths for sources S1 to S5; that is, either all the traffic would flow across Router 1 and Router 5 or across Router 2 and Router 5. It would be impossible in this topology to utilize Router 1 and Router 5 and Router 2 and Router 5 for load splitting. Likewise, the polarization problem would apply to Router 3 and Router 6 and Router 4 and Router 6; that is, it would be impossible in this topology to utilize both Router 3 and Router 6 and Router 4 and Router 6 for load splitting.

ECMP Multicast Load Splitting Based on Source Group and Next-Hop Address

Configuring ECMP multicast load splitting based on source, group, and next-hop address enables a more complex hash, the next-hop-based S-G-hash algorithm, which is based on source, group, and next-hop address. The next-hop-based S-G-hash algorithm is predictable because no randomization is used in calculating the hash value. Unlike the S-hash and basic S-G-hash algorithms, the hash mechanism used by the next-hop-based S-G-hash algorithm is not subject to polarization.

Note The next-hop-based S-G-hash algorithm in IPv4 multicast is the same algorithm used in IPv6 ECMP multicast load splitting, which, in turn, utilizes the same hash function used for PIM-SM bootstrap device (BSR).

The next-hop-based hash mechanism does not produce polarization and also maintains better RPF stability when paths fail. These benefits come at the cost that the source or RP IP addresses cannot be used to reliably predict and engineer the outcome of load splitting when the next-hop-based S-G-hash algorithm is used. Because many customer networks have implemented equal-cost multipath topologies, the manual engineering

of load splitting, thus, is not a requirement in many cases. Rather, it is more of a requirement that the default behavior of IP multicast be similar to IP unicast; that is, it is expected that IP multicast use multiple equal-cost paths on a best-effort basis. Load splitting for IPv4 multicast, therefore, could not be enabled by default because of the anomaly of polarization.

Note Load splitting for CEF unicast also uses a method that does not exhibit polarization and likewise cannot be used to predict the results of load splitting or engineer the outcome of load splitting.

The next-hop-based hash function avoids polarization because it introduces the actual next-hop IP address of PIM neighbors into the calculation, so the hash results are different for each device, and in effect, there is no problem of polarization. In addition to avoiding polarization, this hash mechanism also increases stability of the RPF paths chosen in the face of path failures. Consider a device with four equal-cost paths and a large number of states that are load split across these paths. Suppose that one of these paths fails, leaving only three available paths. With the hash mechanism used by the polarizing hash mechanisms (the hash mechanism used by the S-hash and basic S-G-hash algorithms), the RPF paths of all states would likely reconverge and thus change between those three paths, especially those paths that were already using one of those three paths. These states, therefore, may unnecessarily change their RPF interface and next-hop neighbor. This problem exists simply because the chosen path is determined by taking the total number of paths available into consideration by the algorithm, so once a path changes, the RPF selection for all states is subject to change too. For the next-hop-based hash mechanism, only the states that were using the changed path for RPF would need to reconverge onto one of the three remaining paths. The states that were already using one of those paths would not change. If the fourth path came back up, the states that initially used it would immediately reconverge back to that path without affecting the other states.

Note The next-hop-based S-G-hash algorithm ignores bidir-PIM groups.

Effect of ECMP Multicast Load Splitting on PIM Neighbor Query and Hello Messages for RPF Path Selection

If load splitting of IP multicast traffic over ECMP is not enabled and there are multiple equal-cost paths towards an RP or a source, IPv4 multicast will first elect the highest IP address PIM neighbor. A PIM neighbor is a device from which PIM hello (or PIMv1 query) messages are received. For example, consider a device that has two equal-cost paths learned by an IGP or configured through two static routes. The next hops of these two paths are 10.1.1.1 and 10.1.2.1. If both of these next-hop devices send PIM hello messages, then 10.1.2.1 would be selected as the highest IP address PIM neighbor. If only 10.1.1.1 sends PIM hello messages, then 10.1.1.1 would be selected. If neither of these devices sends PIM hello messages, then 10.1.2.1 would be selected. This deference to PIM hello messages allows the construction of certain types of dynamic failover scenarios with only static multicast routes (mroutes); it is otherwise not very useful.

Note For more information about configuring static mroutes, see the *Configuring Multiple Static Mroutes in Cisco IOS* configuration note on the Cisco IOS IP multicast FTP site, which is available at: ftp://ftpeng.cisco.com/ipmulticast /config-notes/static-mroutes.txt.

When load splitting of IP multicast traffic over ECMP is enabled, the presence of PIM hello message from neighbors is not considered; that is, the chosen RPF neighbor does not depend on whether or not PIM hello

messages are received from that neighbor--it only depends on the presence or absence of an equal-cost route entry.

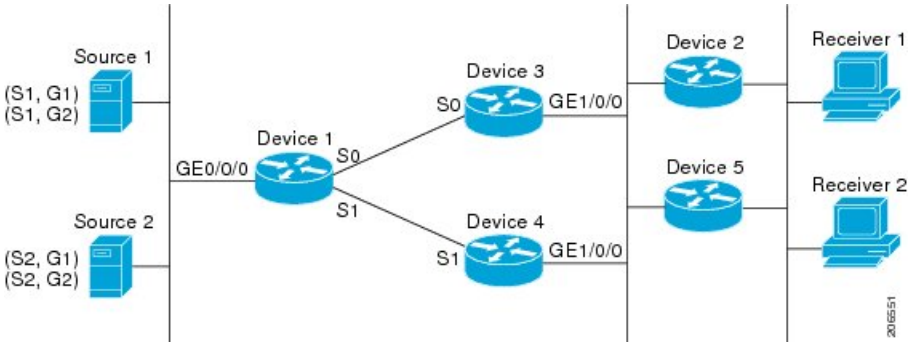
Effect of ECMP Multicast Load Splitting on the PIM Assert Process in PIM-SM and PIM-SSM

There are also cases where ECMP multicast load splitting with the **ip multicast multipath** command can become ineffective due to the PIM assert process taking over, even when using PIM-SM with (*, G) or (S, G) forwarding or PIM-SSM with (S, G) forwarding.

The figure illustrates a sample topology that is used in this section to explain the effect of ECMP multicast load splitting on the PIM assert process in PIM-SM and PIM-SSM.

Note Although the following illustration and example uses routers in the configuration, any device (router or controller) can be used.

Figure 83: ECMP Multicast Load Splitting and the PIM Assert Process in PIM-SM and PIM-SSM



In the topology illustrated in the figure, if both Device 2 and Device 5 are Cisco devices and are consistently configured for ECMP multicast load splitting with the **ip multicast multipath** command, then load splitting would continue to work as expected; that is, both devices would have Device 3 and Device 4 as equal-cost next hops and would sort the list of equal-cost paths in the same way (by IP address). When applying the multipath hash function, for each (S, G) or (*, G) state, they would choose the same RPF neighbor (either Device 3 or Device 4) and send their PIM joins to this neighbor.

If Device 5 and Device 2 are inconsistently configured with the **ip multicast multipath** command, or if Device 5 is a third-party device, then Device 2 and Device 5 may choose different RPF neighbors for some (*, G) or (S, G) states. For example Device 2 could choose Device 3 for a particular (S, G) state or Device 5 could choose Device 4 for a particular (S, G) state. In this scenario, Device 3 and Device 4 would both start to forward traffic for that state onto Gigabit Ethernet interface 1/1, see each other's forwarded traffic, and--to avoid traffic duplication--start the assert process. As a result, for that (S, G) state, the device with the higher IP address for Gigabit Ethernet interface 1/1 would forward the traffic. However, both Device 2 and Device 5 would be tracking the winner of the assert election and would send their PIM joins for that state to this assert winner, even if this assert winner is not the same device as the one that they calculated in their RPF selection. For PIM-SM and PIM-SSM, therefore, the operation of ECMP multicast load splitting can only be guaranteed when all downstream devices on a LAN are consistently configured Cisco devices.

ECMP Multicast Load Splitting and Reconvergence When Unicast Routing Changes

When unicast routing changes, all IP multicast routing states reconverge immediately based on the available unicast routing information. Specifically, if one path goes down, the remaining paths reconverge immediately,

and if the path comes up again, multicast forwarding will subsequently reconverge to the same RPF paths that were used before the path failed. Reconvergence occurs whether load splitting of IP multicast traffic over ECMP is configured or not.

Use of ECMP Multicast Load Splitting with Static Mroutes

If it is not possible to use an IGP to install equal cost routes for certain sources or RPs, static routes can be configured to specify the equal-cost paths for load splitting. You cannot use static mroutes to configure equal-cost paths because the software does not support the configuration of one static mroute per prefix. There are some workarounds for this limitation using recursive route lookups but the workarounds cannot be applied to equal-cost multipath routing.

Note For more information about configuring static mroutes, see the *Configuring Multiple Static Mroutes in Cisco IOS* configuration note on the Cisco IOS IP multicast FTP site at ftp://ftpeng.cisco.com/ipmulticast/config-notes/static-mroutes.txt.

You can specify only static mroutes for equal-cost multipaths in IPv4 multicast; however, those static mroutes would only apply to multicast, or you can specify that the equal-cost multipaths apply to both unicast and multicast routing. In IPv6 multicast, there is no such restriction. Equal-cost multipath mroutes can be configured for static IPv6 mroutes that apply to only unicast routing, only multicast routing, or both unicast and multicast routing.

Alternative Methods of Load Splitting IP Multicast Traffic

Load splitting of IP multicast traffic can also be achieved by consolidating multiple parallel links into a single tunnel over which the multicast traffic is then routed. This method of load splitting is more complex to configure than ECMP multicast load splitting. One such case where configuring load splitting across equal-cost paths using GRE links can be beneficial is the case where the total number of (S, G) or (*, G) states is so small and the bandwidth carried by each state so variable that even the manual engineering of the source or RP addresses cannot guarantee the appropriate load splitting of the traffic.

Note With the availability of ECMP multicast load splitting, tunnels typically only need to be used if per-packet load sharing is required.

IP multicast traffic can also be used to load split across bundle interfaces, such as Gigabit EtherChannel interfaces. GRE or other type of tunnels can also constitute such forms of Layer 2 link bundles. Before using such an Layer 2 mechanism, it is necessary to understand how unicast and multicast traffic is load split.

Before load splitting IP multicast traffic across equal-cost paths over a tunnel, you must configure CEF per-packet load balancing or else the GRE packets will not be load balanced per packet.

How to Load Split IP Multicast Traffic over ECMP

Enabling ECMP Multicast Load Splitting

Perform the following tasks to load split IP multicast traffic across multiple equal-cost paths, based on source address.

If two or more equal-cost paths from a source are available, unicast traffic will be load split across those paths. However, by default, multicast traffic is not load split across multiple equal-cost paths. In general, multicast traffic flows down from the RPF neighbor. According to PIM specifications, this neighbor must have the highest IP address if more than one neighbor has the same metric.

Configuring load splitting with the **ip multicast multipath** command causes the system to load split multicast traffic across multiple equal-cost paths based on source address using the S-hash algorithm. When the **ip multicast multipath** command is configured and multiple equal-cost paths exist, the path in which multicast traffic will travel is selected based on the source IP address. Multicast traffic from different sources will be load split across the different equal-cost paths. Load splitting will not occur across equal-cost paths for multicast traffic from the same source sent to different multicast groups.

Note The **ip multicast multipath**command load splits the traffic and does not load balance the traffic. Traffic from a source will use only one path, even if the traffic far outweighs traffic from other sources.

Prerequisites for IP Multicast Load Splitting - ECMP

- You must have an adequate number of sources (at least more than two sources) to enable ECMP multicast load splitting based on source address.
- You must have multiple paths available to the RP to configure ECMP multicast load splitting.

Note Use the **show ip route** command with either the IP address of the source for the *ip-address*argument or the IP address of the RP to validate that there are multiple paths available to the source or RP, respectively. If you do not see multiple paths in the output of the command, you will not be able to configure ECMP multicast load splitting.

- When using PIM-SM with shortest path tree (SPT) forwarding, the T-bit mus be set for the forwarding of all (S, G) states.
- Before configuring ECMP multicast load splitting, it is best practice to use the **show ip rpf** command to validate whether sources can take advantage of IP multicast multipath capabilities.

Restrictions for IP Multicast Load Splitting -ECMP

- If two or more equal-cost paths from a source are available, unicast traffic will be load split across those paths. However, by default, multicast traffic is not load split across multiple equal-cost paths. In general,

multicast traffic flows down from the RPF neighbor. According to PIM specifications, this neighbor must have the highest IP address if more than one neighbor has the same metric.

- The **ip multicast multipath** command does not support configurations in which the same PIM neighbor IP address is reachable through multiple equal-cost paths. This situation typically occurs if unnumbered interfaces are used. Use different IP addresses for all interfaces when configuring the **ip multicast multipath** command.
- The **ip multicast multipath** command load splits the traffic and does not load balance the traffic. Traffic from a source will use only one path, even if the traffic far outweighs traffic from other sources.

Enabling ECMP Multicast Load Splitting Based on Source Address

Perform this task to enable ECMP multicast load splitting of multicast traffic based on source address (using the S-hash algorithm) to take advantage of multiple paths through the network. The S-hash algorithm is predictable because no randomization is used in calculating the hash value. The S-hash algorithm, however, is subject to polarization because for a given source, the same hash is always picked irrespective of the device on which the hash is being calculated.

Note Enable ECMP multicast load splitting on the device that is to be the receiver for traffic from more than one incoming interfaces, which is opposite to unicast routing. From the perspective of unicast, multicast is active on the sending device connecting to more than one outgoing interfaces.

Before you begin

- You must have an adequate number of sources (at least more than two sources) to enable ECMP multicast load splitting based on source address.
- You must have multiple paths available to the RP to configure ECMP multicast load splitting.

Note Use the **show ip route** command with either the IP address of the source for the *ip-address*argument or the IP address of the RP to validate that there are multiple paths available to the source or RP, respectively. If you do not see multiple paths in the output of the command, you will not be able to configure ECMP multicast load splitting.

- When using PIM-SM with shortest path tree (SPT) forwarding, the T-bit mus be set for the forwarding of all (S, G) states.
- Before configuring ECMP multicast load splitting, it is best practice to use the **show ip rpf** command to validate whether sources can take advantage of IP multicast multipath capabilities.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip multicast multipath** **Example:** `Device(config)# ip multicast multipath`	Enables ECMP multicast load splitting based on source address using the S-hash algorithm. • Because this command changes the way an RPF neighbor is selected, it must be configured consistently on all devices in a redundant topology to avoid looping. • This command does not support configurations in which the same PIM neighbor IP address is reachable through multiple equal-cost paths. This situation typically occurs if unnumbered interfaces are used. Use a different IP address for each interface in a device on which this command is to be configured. • This command load splits the traffic and does not load balance the traffic. Traffic from a source will use only one path, even if the traffic far outweighs traffic from other sources.
Step 4	Repeat step 3 on all the devices in a redundant topology.	--
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	**show ip rpf** *source-address* [*group-address*] **Example:** `Device# show ip rpf 10.1.1.2`	(Optional) Displays the information that IP multicast routing uses to perform the RPF check. • Use this command to verify RPF selection so as to ensure that IP multicast traffic is being properly load split.
Step 7	**show ip route** *ip-address* **Example:**	(Optional) Displays the current state of the IP routing table.

	Command or Action	Purpose
	`Device# show ip route 10.1.1.2`	• Use this command to verify that there multiple paths available to a source or RP for ECMP multicast load splitting. • For the *ip-address* argument, enter the IP address of a source to validate that there are multiple paths available to the source (for shortest path trees) or the IP address of an RP to validate that there are multiple paths available to the RP (for shared trees).

Enabling ECMP Multicast Load Splitting Based on Source and Group Address

Perform this task to enable ECMP multicast load splitting of multicast traffic based on source and group address (using the basic S-G-hash algorithm) to take advantage of multiple paths through the network. The basic S-G-hash algorithm is predictable because no randomization is used in calculating the hash value.The basic S-G-hash algorithm, however, is subject to polarization because for a given source and group, the same hash is always picked irrespective of the device on which the hash is being calculated.

The basic S-G-hash algorithm provides more flexible support for ECMP multicast load splitting than the the S-hash algorithm. Using the basic S-G-hash algorithm for load splitting, in particular, enables multicast traffic from devices that send many streams to groups or that broadcast many channels, such as IPTV servers or MPEG video servers, to be more effectively load split across equal-cost paths.

Note Enable ECMP multicast load splitting on the device that is to be the receiver for traffic from more than one incoming interfaces, which is opposite to unicast routing. From the perspective of unicast, multicast is active on the sending device connecting to more than one outgoing interfaces.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip multicast multipath s-g-hash basic** **Example:** `Device(config)# ip multicast multipath s-g-hash basic`	Enables ECMP multicast load splitting based on source and group address using the basic S-G-hash algorithm. • Because this command changes the way an RPF neighbor is selected, it must be configured consistently on all devices in a redundant topology to avoid looping.
Step 4	Repeat Step 3 on all the devices in a redundant topology.	--
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	**show ip rpf** *source-address* [*group-address*] **Example:** `Device# show ip rpf 10.1.1.2`	(Optional) Displays the information that IP multicast routing uses to perform the RPF check. • Use this command to verify RPF selection so as to ensure that IP multicast traffic is being properly load split.
Step 7	**show ip route** *ip-address* **Example:** `Device# show ip route 10.1.1.2`	(Optional) Displays the current state of the IP routing table. • Use this command to verify that there multiple paths available to a source or RP for ECMP multicast load splitting. • For the *ip-address* argument, enter the IP address of a source to validate that there are multiple paths available to the source (for shortest path trees) or the IP address of an RP to validate that there are multiple paths available to the RP (for shared trees).

Enabling ECMP Multicast Load Splitting Based on Source Group and Next-Hop Address

Perform this task to enable ECMP multicast load splitting of multicast traffic based on source, group, and next-hop address (using the next-hop-based S-G-hash algorithm) to take advantage of multiple paths through the network. The next-hop-based S-G-hash algorithm is predictable because no randomization is used in calculating the hash value. Unlike the S-hash and basic S-G-hash algorithms, the hash mechanism used by the next-hop-based S-G-hash algorithm is not subject to polarization.

The next-hop-based S-G-hash algorithm provides more flexible support for ECMP multicast load splitting than S-hash algorithm and eliminates the polarization problem. Using the next-hop-based S-G-hash algorithm for ECMP multicast load splitting enables multicast traffic from devices that send many streams to groups or

that broadcast many channels, such as IPTV servers or MPEG video servers, to be more effectively load split across equal-cost paths.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip multicast multipath s-g-hash next-hop-based** **Example:** `Device(config)# ip multicast multipath s-g-hash next-hop-based`	Enables ECMP multicast load splitting based on source, group, and next-hop-address using the next-hop-based S-G-hash algorithm. • Because this command changes the way an RPF neighbor is selected, it must be configured consistently on all routers in a redundant topology to avoid looping. **Note** Be sure to enable the **ip multicast multipath** command on the router that is supposed to be the receiver for traffic from more than one incoming interfaces, which is opposite to unicast routing. From the perspective of unicast, multicast is active on the sending router connecting to more than one outgoing interfaces.
Step 4	Repeat Steps 1 through 3 on all the routers in a redundant topology.	--
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	**show ip rpf** *source-address* [*group-address*] **Example:** `Device# show ip rpf 10.1.1.2`	(Optional) Displays the information that IP multicast routing uses to perform the RPF check. • Use this command to verify RPF selection so as to ensure that IP multicast traffic is being properly load split.

	Command or Action	Purpose
Step 7	**show ip route** *ip-address* **Example:** `Device# show ip route 10.1.1.2`	(Optional) Displays the current state of the IP routing table. • Use this command to verify that there multiple paths available to a source or RP for ECMP multicast load splitting. • For the *ip-address* argument, enter the IP address of a source to validate that there are multiple paths available to the source (for shortest path trees) or the IP address of an RP to validate that there are multiple paths available to the RP (for shared trees).

Configuration Examples for Load Splitting IP Multicast Traffic over ECMP

Example Enabling ECMP Multicast Load Splitting Based on Source Address

The following example shows how to enable ECMP multicast load splitting on a router based on source address using the S-hash algorithm:

```
ip multicast multipath
```

Example Enabling ECMP Multicast Load Splitting Based on Source and Group Address

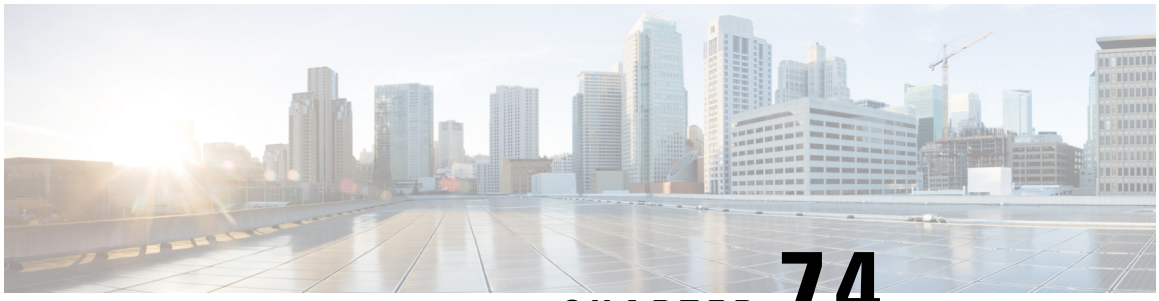
The following example shows how to enable ECMP multicast load splitting on a router based on source and group address using the basic S-G-hash algorithm:

```
ip multicast multipath s-g-hash basic
```

Example Enabling ECMP Multicast Load Splitting Based on Source Group and Next-Hop Address

The following example shows how to enable ECMP multicast load splitting on a router based on source, group, and next-hop address using the next-hop-based S-G-hash algorithm:

```
ip multicast multipath s-g-hash next-hop-based
```



CHAPTER 74

IP Multicast Optimization: SSM Channel Based Filtering for Multicast

Prerequisites for SSM Channel Based Filtering for Multicast Boundaries

IP multicast is enabled on the device using the tasks described in the "Configuring Basic IP Multicast" module of the *IP Multicast: PIM Configuration Guide*.

Information About the SSM Channel Based Filtering for Multicast Boundaries

This section provides information about the SSM channel based filtering for multicast boundaries feature.

Rules for Multicast Boundaries

The SSM Channel Based Filtering for Multicast Boundaries feature expands the **ip multicast boundary** command for control plane filtering support. More than one **ip multicast boundary** command can be applied to an interface.

The following rules govern the **ip multicast boundary** command:

- One instance of the **in** and **out** keywords can be configured on an interface.
- The **in** and **out** keywords can be used for standard or extended access lists.
- Only standard access lists are permitted with the use of the **filter-autorp** keyword or no keyword.

- A maximum of three instances of a command will be allowed on an interface: one instance of **in**, one instance of **out**, and one instance of **filter-autorp** or no keyword.
- When multiple instances of the command are used, the filtering will be cumulative. If a boundary statement with no keyword exists with a boundary statement with the **in**keyword, both access lists will be applied on the in direction and a match on either one will be sufficient.
- All instances of the command apply to both control and data plane traffic.
- Protocol information on the extended access list is parsed to allow reuse and filtering for consistency. An (S,G) operation will be filtered by an extended access list under all conditions stated above for keywords if the access list filters (S,G) traffic for all protocols.

Benefits of SSM Channel Based Filtering for Multicast Boundaries

- This feature allows input on the source interface.
- The access control capabilities are the same for SSM and Any Source Multicast (ASM).

How to Configure SSM Channel Based Filtering for Multicast Boundaries

This section provides steps for configuring SSM channel based filtering for multicast boundaries.

Configuring Multicast Boundaries

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip access-list** {**standard**| **extended**} *access-list-name* **Example:** `Device(config)# ip access-list 101`	Configures the standard or extended access list.

	Command or Action	Purpose
Step 4	**permit** *protocol* **host** *address* **host** *address* **Example:** `Device(config-ext-nacl)# permit ip host 181.1.2.201 host 232.1.1.11`	Permits specified ip host traffic.
Step 5	**deny** *protocol* **host** *address* **host** *address* **Example:** `Device(config-acl-nacl)# deny ip host 181.1.2.203 host 232.1.1.1`	Denies specified multicast ip group and source traffic.
Step 6	Repeat Step 4 or Step 5 as needed.	Permits and denies specified host and source traffic.
Step 7	**interface** *type* **interface-number** *port-number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enables interface configuration mode.
Step 8	**ip multicast boundary** *access-list-name* [**in**| **out** | **filter-autorp**] **Example:** `Device(config-if)# ip multicast boundary acc_grp1 out`	Configures the multicast boundary. **Note** The **filter-autorp** keyword does not support extended access lists.

Configuration Examples for SSM Channel Based Filtering for Multicast Boundaries

This section provides configuration examples of SSM Channel Based filtering for multicast boundaries.

Configuring the Multicast Boundaries Permitting and Denying Traffic Example

The following example permits outgoing traffic for (181.1.2.201, 232.1.1.1) and (181.1.2.202, 232.1.1.1) and denies all other (S,G)s.

```
configure terminal
 ip access-list extended acc_grp1
 permit ip host 0.0.0.0 232.1.1.1 0.0.0.255
 permit ip host 181.1.2.201 host 232.1.1.1
 permit udp host 181.1.2.202 host 232.1.1.1
 permit ip host 181.1.2.202 host 232.1.1.1
```

```
 deny igmp host 181.2.3.303 host 232.1.1.1
 interface gigabitethernet 1/1
 ip multicast boundary acc_grp1 out
```

Configuring the Multicast Boundaries Permitting Traffic Example

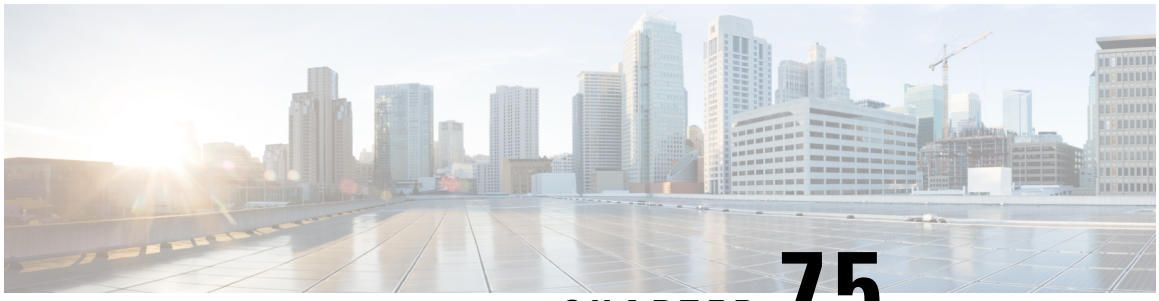
The following example permits outgoing traffic for (192.168.2.201, 232.1.1.5) and 192.168.2.202, 232.1.1.5).

```
configure terminal
ip access-list extended acc_grp6
 permit ip host 0.0.0.0 232.1.1.1 5.0.0.255
 deny udp host 192.168.2.201 host 232.1.1.5
 permit ip host 192.168.2.201 host 232.1.1.5
 deny pim host 192.168.2.201 host 232.1.1.5
 permit ip host 192.168.2.202 host 232.1.1.5
 deny igmp host 192.2.3.303 host 232.1.1.1
interface gigabitethernet 1/1
 ip multicast boundary acc_grp6 out
```

Configuring the Multicast Boundaries Denying Traffic Example

The following example denies a group-range that is announced by the candidate RP. Because the group range is denied, no pim auto-rp mappings are created.

```
configure terminal
ip access-list standard acc_grp10
 deny 225.0.0.0 0.255.255.255
 permit any
 access-list extended acc_grp12
 permit pim host 181.1.2.201 host 232.1.1.8
 deny udp host 181.1.2.201 host 232.1.1.8
 permit pim host 181.1.2.203 0.0.0.255 host 227.7.7.7
 permit ip host 0.0.0.0 host 227.7.7.7
 permit ip 181.1.2.203 0.0.0.255 host 227.7.7.7
 permit ip host 181.1.2.201 host 232.1.1.7
ip access-list extended acc_grp13
 deny ip host 181.1.2.201 host 232.1.1.8
 permit ip any any
interface gigabitethernet 1/1
 ip multicast boundary acc_grp10 filter-autorp
 ip multicast boundary acc_grp12 out
 ip multicast boundary acc_grp13 in
```



CHAPTER 75

IP Multicast Optimization: IGMP State Limit

Prerequisites for IGMP State Limit

- IP multicast is enabled and the Protocol Independent Multicast (PIM) interfaces are configured using the tasks described in the "Configuring Basic IP Multicast" module of the *IP Multicast: PIM Configuration Guide*.
- ALL ACLs must be configured. For information, see the see the " Creating an IP Access List and Applying It to an Interface " module of the *Security Configuration Guide: Access Control Lists* guide.

Restrictions for IGMP State Limit

You can configure only one global limit per device and one limit per interface.

Information About IGMP State Limit

This section provides information about IGMP state limit.

IGMP State Limit

The IGMP State Limit feature allows for the configuration of IGMP state limiters, which impose limits on mroute states resulting from IGMP membership reports (IGMP joins) on a global or per interface basis. Membership reports exceeding the configured limits are not entered into the IGMP cache. This feature can be used to prevent DoS attacks or to provide a multicast CAC mechanism in network environments where all the multicast flows roughly utilize the same amount of bandwidth.

Note IGMP state limiters impose limits on the number of mroute states resulting from IGMP, IGMP v3lite, and URL Rendezvous Directory (URD) membership reports on a global or per interface basis.

IGMP State Limit Feature Design

- Configuring IGMP state limiters in global configuration mode specifies a global limit on the number of IGMP membership reports that can be cached.
- Configuring IGMP state limiters in interface configuration mode specifies a limit on the number of IGMP membership reports on a per interface basis.
- Use ACLs to prevent groups or channels from being counted against the interface limit. A standard or an extended ACL can be specified. A standard ACL can be used to define the (*, G) state to be excluded from the limit on an interface. An extended ACLs can be used to define the (S, G) state to be excluded from the limit on an interface. An extended ACL also can be used to define the (*, G) state to be excluded from the limit on an interface, by specifying 0.0.0.0 for the source address and source wildcard--referred to as (0, G)--in the permit or deny statements that compose the extended access list.
- You can only configure one global limit per device and one limit per interface.

Mechanics of IGMP State Limiters

The mechanics of IGMP state limiters are as follows:

- Each time a router receives an IGMP membership report for a particular group or channel, the Cisco IOS software checks to see if either the limit for the global IGMP state limiter or the limit for the per interface IGMP state limiter has been reached.
- If only a global IGMP state limiter has been configured and the limit has not been reached, IGMP membership reports are honored. When the configured limit has been reached, subsequent IGMP membership reports are then ignored (dropped) and a warning message in one of the following formats is generated:
  -
    ```
    %IGMP-6-IGMP_GROUP_LIMIT: IGMP limit exceeded for <group (*, group address)> on
    <interface type number> by host <ip address>
    ```
  -
    ```
    %IGMP-6-IGMP_CHANNEL_LIMIT: IGMP limit exceeded for <channel (source address, group
     address)> on <interface type number> by host <ip address>
    ```
- If only per interface IGMP state limiters are configured, then each limit is only counted against the interface on which it was configured.
- If both a global IGMP state limiter and per interface IGMP state limiters are configured, the limits configured for the per interface IGMP state limiters are still enforced but are constrained by the global limit.

How to Configure IGMP State Limit

This section describes how to configure IGMP state limit.

Configuring IGMP State Limiters

IGMP state limiters impose limits on the number of mroute states resulting from IGMP, IGMP v3lite, and URD membership reports on a global or per interface basis.

Configuring Global IGMP State Limiters

Perform this optional task to configure one global IGMP state limiter per device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip igmp limit** *number* **Example:** `Device(config)# ip igmp limit 150`	Configures a global limit on the number of mroute states resulting from IGMP membership reports (IGMP joins).
Step 4	**end** **Example:** `Device(config-if)# end`	Ends the current configuration session and returns to privileged EXEC mode.
Step 5	**show ip igmp groups** **Example:** `Device# show ip igmp groups`	(Optional) Displays the multicast groups with receivers that are directly connected to the device and that were learned through IGMP.

Configuring Per Interface IGMP State Limiters

Perform this optional task to configure a per interface IGMP state limiter.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface Gigabitethernet 1/1`	Enters interface configuration mode. • Specify an interface that is connected to hosts.
Step 4	**ip igmp limit** *number* [**except** *access-list*] **Example:** `Device(config-if)# ip igmp limit 100`	Configures a per interface limit on the number of mroutes states created as a result of IGMP membership reports (IGMP joins).
Step 5	Do one of the following: • **exit** • **end** **Example:** `Device(config-if)# exit` `Device(config-if)# end`	• (Optional) Ends the current configuration session and returns to global configuration mode. Repeat steps 3 and 4 to configure a per interface limiter on another interface. • Ends the current configuration session and returns to privileged EXEC mode.
Step 6	**show ip igmp interface** [*type number*] **Example:** `Device# show ip igmp interface`	(Optional) Displays information about the status and configuration of IGMP and multicast routing on interfaces.
Step 7	**show ip igmp groups** **Example:** `Device# show ip igmp groups`	(Optional) Displays the multicast groups with receivers that are directly connected to the device and that were learned through IGMP.

Configuration examples for IGMP State Limit

This section show configuration examples of IGMP state limit.

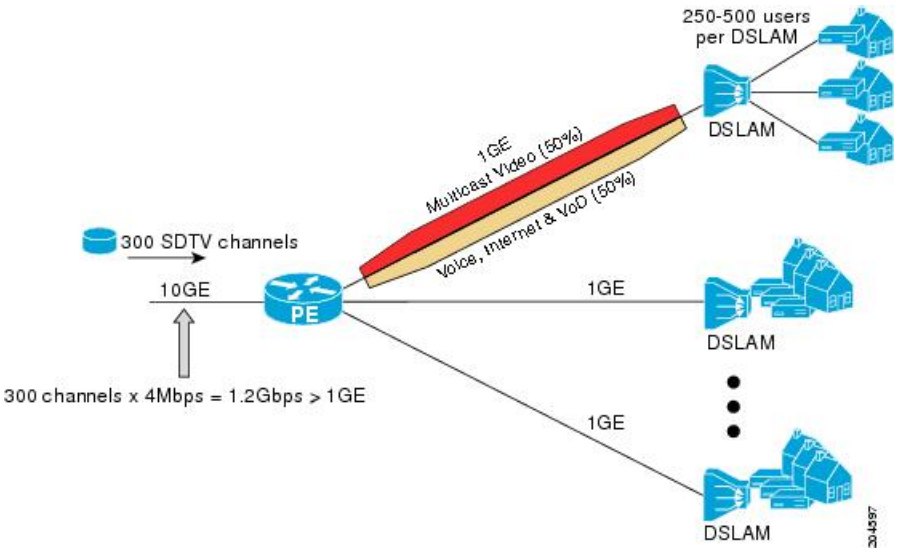
Configuring IGMP State Limiters Example

The following example shows how to configure IGMP state limiters to provide multicast CAC in a network environment where all the multicast flows roughly utilize the same amount of bandwidth.

This example uses the topology illustrated in the figure.

Note Although the following illustration and example uses routers in the configuration, any device (router or switch) can be used.

Figure 84: IGMP State Limit Example Topology



In this example, a service provider is offering 300 Standard Definition (SD) TV channels. Each SD channel utilizes approximately 4 Mbps.

The service provider must provision the Gigabit Ethernet interfaces on the PE router connected to the Digital Subscriber Line Access Multiplexers (DSLAMs) as follows: 50% of the link's bandwidth (500 Mbps) must be available to subscribers of the Internet, voice, and video on demand (VoD) service offerings while the remaining 50% (500 Mbps) of the link's bandwidth must be available to subscribers of the SD channel offerings.

Because each SD channel utilizes the same amount of bandwidth (4 Mbps), per interface IGMP state limiters can be used to provide the necessary CAC to provision the services being offered by the service provider. To determine the required CAC needed per interface, the total number of channels is divided by 4 (because each channel utilizes 4 Mbps of bandwidth). The required CAC needed per interface, therefore, is as follows:

500Mbps / 4Mbps = 125 mroutes

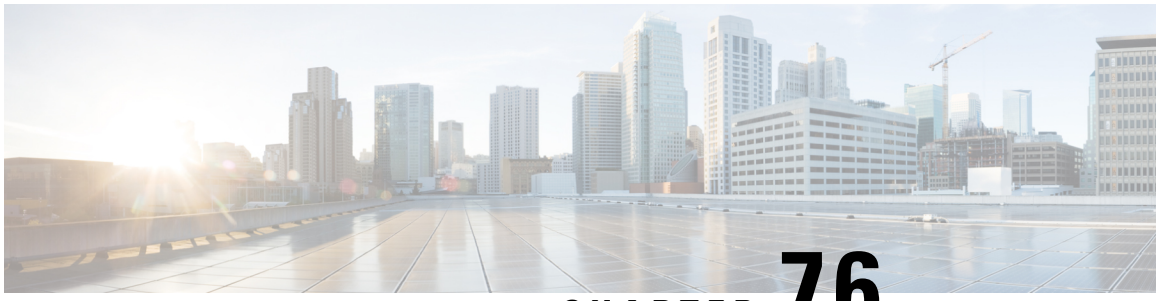
Once the required CAC is determined, the service provider uses the results to configure the per IGMP state limiters required to provision the Gigabit Ethernet interfaces on the PE router. Based on the network's CAC requirements, the service provider must limit the SD channels that can be transmitted out a Gigabit Ethernet interface (at any given time) to 125. Configuring a per interface IGMP state limit of 125 for the SD channels provisions the interface for 500 Mbps of bandwidth, the 50% of the link's bandwidth that must always be available (but never exceeded) for the SD channel offerings.

The following configuration shows how the service provider uses a per interface mroute state limiter to provision interface Gigabit Ethernet 1/1 for the SD channels and Internet, Voice, and VoD services being offered to subscribers:

```
interface GigabitEthernet1/1
description --- Interface towards the DSLAM ---
.
.
.
ip igmp limit 125
```

PART IV

Security



CHAPTER 76

Controlling Switch Access with Passwords and Privilege Levels

Restrictions for Controlling Switch Access with Passwords and Privileges

The following are the restrictions for controlling switch access with passwords and privileges:

- Disabling password recovery will not work if you have set the switch to boot up manually by using the **boot manual** global configuration command. This command produces the boot loader prompt (*switch:*) after the switch is power cycled.
- Password validation for the **enable password** command against the common criteria policy does not happen during configuration or reconfiguration of the **aaa common-criteria policy** command. The password is validated against the common criteria policy only during configuration or reconfiguration of the **enable common-criteria-policy** command.

Restrictions and Guidelines for Reversible Password Types

- Password type 0 and 7 are replaced with password type 6. So password type 0 and 7, which were used for administrator login to the console, Telnet, SSH, webUI, and NETCONF must be migrated to password type 6. No action is required if username and password are type 0 and 7 for local authentication such as CHAP, EAP, and so on.
- If the startup configuration has a type 6 password and you downgrade to a version in which type 6 password is not supported, you can/may be locked out of the device.

Restrictions and Guidelines for Irreversible Password Types

- Username secret password type 5 and enable secret password type 5 must be migrated to the stronger password type 8 or 9. For more information, see Protecting Enable and Enable Secret Passwords with Encryption, on page 1065.
- If the startup configuration of the device has convoluted type 9 secret (password that starts with 14), then a downgrade can only be performed to a release in which the convoluted type 9 secret is supported.

  Before you downgrade to any release in which convoluted type 9 secret is not supported, ensure that the type 9 secret (password that starts with 9) must be part of the startup configuration instead of convoluted type 9 secret (password that starts with 14) or type 5 secret (password that starts with 1).
- Plain text passwords are converted to nonreversible encrypted password type 9.
- Secret password type 4 is not supported.

Information About Controlling Switch Access with Passwords and Privileges

This section provides information about controlling switch access with passwords and privileges.

Preventing Unauthorized Access

You can prevent unauthorized users from reconfiguring your switch and viewing configuration information. Typically, you want network administrators to have access to your switch while you restrict access to users who dial from outside the network through an asynchronous port, connect from outside the network through a serial port, or connect through a terminal or workstation from within the local network.

To prevent unauthorized access into your switch, you should configure one or more of these security features:

- At a minimum, you should configure passwords and privileges at each switch port. These passwords are locally stored on the switch. When users attempt to access the switch through a port or line, they must enter the password specified for the port or line before they can access the switch.
- For an additional layer of security, you can also configure username and password pairs, which are locally stored on the switch. These pairs are assigned to lines or ports and authenticate each user before that user can access the switch. If you have defined privilege levels, you can also assign a specific privilege level (with associated rights and privileges) to each username and password pair.
- If you want to use username and password pairs, but you want to store them centrally on a server instead of locally, you can store them in a database on a security server. Multiple networking devices can then use the same database to obtain user authentication (and, if necessary, authorization) information.
- You can also enable the login enhancements feature, which logs both failed and unsuccessful login attempts. Login enhancements can also be configured to block future login attempts after a set number of unsuccessful attempts are made.

Default Password and Privilege Level Configuration

A simple way of providing terminal access control in your network is to use passwords and assign privilege levels. Password protection restricts access to a network or network device. Privilege levels define what commands users can enter after they have logged into a network device.

This table shows the default password and privilege level configuration.

Table 85: Default Password and Privilege Levels

Feature	Default Setting
Enable password and privilege level	No password is defined. The default is level 15 (privileged EXEC level). The password is not encrypted in the configuration file.
Enable secret password and privilege level	No password is defined. The default is level 15 (privileged EXEC level). The password is encrypted before it is written to the configuration file.
Line password	No password is defined.

Additional Password Security

The following sections provide information about unmasked and masked secret password.

Unmasked Secret Password

To provide an additional layer of security, particularly for passwords that cross the network or that are stored on a Trivial File Transfer Protocol (TFTP) server, you can use either the **enable password** or **enable secret** global configuration commands. Both commands accomplish the same thing; that is, you can establish an encrypted password that users must enter to access privileged EXEC mode (the default) or any privilege level you specify.

We recommend that you use the **enable secret** command because it uses an improved encryption algorithm. If you configure the **enable secret** command, it takes precedence over the **enable password** command; the two commands cannot be in effect simultaneously.

For a device that loads with no start-up configuration, the Enable Secret Password task is a mandatory configuration whether you select **Yes** or **No** at the `"Would you like to enter the initial configuration dialog?"` prompt of the initial configuration wizard. The configured password must contain a minimum of 10 and a maximum of 32 characters. It must also include a minimum of one uppercase letter, one lowercase letter, and one numeral. Additionally, the term 'cisco' must not be part of the password.

Note In some cases where the device is connected to the internet, Cisco Plug and Play (PnP) can terminate the initial configuration wizard. In such cases, the enable secret configuration will not be prompted.

If you enable password encryption, it applies to all passwords including username passwords, authentication key passwords, the privileged command password, and console and virtual terminal line passwords.

Masked Secret Password

With **enable secret** command, password is encrypted but is visible on the terminal when you type the password. To mask the password on the terminal, use the **masked-secret** global configuration command. The encryption type for this password is type 9, by default.

You can use this command to configure masked secret password for common criteria policy.

Password Recovery

By default, any end user with physical access to the switch can recover from a lost password by interrupting the boot process while the switch is powering on and then by entering a new password.

The password-recovery disable feature protects access to the switch password by disabling part of this functionality. When this feature is enabled, the end user can interrupt the boot process only by agreeing to set the system back to the default configuration. With password recovery disabled, you can still interrupt the boot process and change the password, but the configuration file (config.text) and the VLAN database file (vlan.dat) are deleted.

If you disable password recovery, we recommend that you keep a backup copy of the configuration file on a secure server in case the end user interrupts the boot process and sets the system back to default values. Do not keep a backup copy of the configuration file on the switch. If the switch is operating in VTP transparent mode, we recommend that you also keep a backup copy of the VLAN database file on a secure server. When the switch is returned to the default system configuration, you can download the saved files to the switch by using the Xmodem protocol.

To re-enable password recovery, use the **no system disable password recovery switch** *number* | *all* global configuration command.

Terminal Line Telnet Configuration

When you power-up your switch for the first time, an automatic setup program runs to assign IP information and to create a default configuration for continued use. The setup program also prompts you to configure your switch for Telnet access through a password. If you did not configure this password during the setup program, you can configure it when you set a Telnet password for a terminal line.

Username and Password Pairs

You can configure username and password pairs, which are locally stored on the switch. These pairs are assigned to lines or ports and authenticate each user before that user can access the switch. If you have defined privilege levels, you can also assign a specific privilege level (with associated rights and privileges) to each username and password pair.

Privilege Levels

Cisco devices use privilege levels to provide password security for different levels of switch operation. By default, the Cisco IOS XE software operates in two modes (privilege levels) of password security: user EXEC (Level 1) and privileged EXEC (Level 15). You can configure up to 16 hierarchical levels of commands for each mode. By configuring multiple passwords, you can allow different sets of users to have access to specified commands.

Privilege Levels on Lines

Users can override the privilege level you set using the **privilege level** line configuration command by logging in to the line and enabling a different privilege level. They can lower the privilege level by using the **disable** command. If users know the password to a higher privilege level, they can use that password to enable the higher privilege level. You might specify a high level or privilege level for your console line to restrict line usage.

For example, if you want many users to have access to the **clear line** command, you can assign it level 2 security and distribute the level 2 password fairly widely. But if you want more restricted access to the **configure** command, you can assign it level 3 security and distribute that password to a more restricted group of users.

Command Privilege Levels

When you set a command to a privilege level, all commands whose syntax is a subset of that command are also set to that level. For example, if you set the **show ip traffic** command to level 15, the **show** commands and **show ip** commands are automatically set to privilege level 15 unless you set them individually to different levels.

AES Password Encryption and Master Encryption Keys

You can enable strong, reversible 128-bit Advanced Encryption Standard (AES) password encryption, also known as type 6 encryption. To start using type 6 encryption, enable the AES Password Encryption feature and configure a master encryption key to encrypt and decrypt passwords.

After you enable AES password encryption and configure a master key, all the existing and newly created cleartext passwords for the supported applications are stored in type 6 encrypted format, unless you disable type 6 password encryption. You can also configure the device to convert all the existing weakly encrypted passwords to type 6 encrypted passwords.

Type 6 encrypted password that is configured must be compatible with the master key existing on the private NVRAM of the device. If it is not compatible, the configuration fails.

Type 0 and 7 passwords can be autoconverted to type 6 if the AES Password Encryption feature and master encryption key are configured.

Note

- Type 6 encrypted password for the username password and autoconversion to password type 6 are supported
- Type 6 encrypted password for the username password is backward compatible. After autoconversion, to prevent an administrator password from getting rejected during a downgrade, migrate the passwords used for administrator logins (management access) to irreversible password types manually.
- Type 6 encrypted password for enable password, autoconversion to password type 6, and line VTY password are supported.

How to Configure Switch Access with Passwords and Privileges

The following sections provide information about the various tasks to access the switch with passwords and privileges.

Setting or Changing a Static Enable Password

The enable password controls access to the privileged EXEC mode. Follow these steps to set or change a static enable password:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**enable** [**common-criteria-policy** *policy-name*] **password** *password* **Example:** `Device(config)# enable password secret321`	Defines a new password or changes an existing password for access to privileged EXEC mode. • By default, no password is defined. • For *policy-name*, specify a policy name defined using the **aaa common-criteria policy** command. **Note** **aaa new-model** and **aaa common-criteria policy** commands must be configured before attaching the **common-criteria-policy** option to the password. • For *password*, specify a string from 1 to 25 alphanumeric characters. The string cannot start with a number, is case sensitive, and allows spaces, but ignores leading spaces. It can contain the question mark (?) character if you precede the question mark with the key combination Crtl-v when you create the password. For example, to create the password abc?123, do this: **a.** Enter **abc**. **b.** Enter **Crtl-v**. **c.** Enter **?123**. When the system prompts you to enter the enable password, you need not precede the

	Command or Action	Purpose
		question mark with Ctrl-v; you can simply enter **abc?123** at the password prompt.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Protecting Enable and Enable Secret Passwords with Encryption

Follow these steps to establish an encrypted password that users must enter to access privileged EXEC mode (the default) or any privilege level you specify:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	Use one of the following: • **enable password** [**level** *level*] {*unencrypted-password* | *encryption-type encrypted-password*} • **enable secret** [**level** *level*] {*unencrypted-password* | *encryption-type encrypted-password*} **Example:** `Device(config)# enable password level 12 example123` or `Device(config)# enable secret 9 $9$sMLBsTFXLnnHTk$0L82`	• Defines a new password or changes an existing password for access to privileged EXEC mode. • Defines a secret password, which is saved using a nonreversible encryption method. • (Optional) For *level*, the range is from 0 to 15. Level 1 is normal user EXEC mode privileges. The default level is 15 (privileged EXEC mode privileges). • For *unencrypted-password*, specify a string from 1 to 25 alphanumeric characters. The string cannot start with a number, is case sensitive, and allows spaces but ignores leading spaces. By default, no password is defined. • For *encryption-type*, the available options for **enable password** are type 0, 6, and 7, and type 0, 5, 8, and 9 for

	Command or Action	Purpose
		enable secret. If you specify an encryption type, you must provide an encrypted password—an encrypted password that you copy from another switch configuration. Secret encryption type 9 is more secure, so we recommend that you select type 9 to avoid any issues while upgrading or downgrading. **Note** • If you do not specify an encryption type for the secret password, the password is auto converted to type 9. • If you specify an encryption type and then enter a clear text password, it will result in an error. • You can also configure type 9 encryption for the secret password manually by using the **algorithm-type scrypt** command in global configuration mode. For example: `Device(config)# username user1 algorithm-type scrypt secret cisco` Or `Device(config)# enable algorithm-type scrypt secret cisco` Run the **write memory** command in privileged EXEC mode for the type 9 secret to be permanently written into the startup configuration.
Step 4	**service password-encryption** **Example:** `Device(config)# service password-encryption`	(Optional) Encrypts the password when the password is defined or when the configuration is written. Encryption prevents the password from being readable in the configuration file.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Disabling Password Recovery

Follow these steps to disable password recovery to protect the security of your switch:

Before you begin

If you disable password recovery, we recommend that you keep a backup copy of the configuration file on a secure server in case the end user interrupts the boot process and sets the system back to default values. Do not keep a backup copy of the configuration file on the switch. If the switch is operating in VTP transparent mode, we recommend that you also keep a backup copy of the VLAN database file on a secure server. When the switch is returned to the default system configuration, you can download the saved files to the switch by using the Xmodem protocol.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**system disable password recovery switch** {*all* | *<1-1>*} **Example:** `Device(config)# system disable password recovery switch all`	Disables password recovery. • *all*: Sets the configuration on switches. • *<1-1>*: Switch number. This setting is saved in an area of the flash memory that is accessible by the boot loader and the Cisco IOS image, but is not a part of the file system and is not accessible by any user.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

What to do next

To remove **disable password recovery**, use the **no system disable password recovery switch all** global configuration command.

Setting a Telnet Password for a Terminal Line

Beginning in user EXEC mode, follow these steps to set a Telnet password for the connected terminal line:

Before you begin

- Attach a PC or workstation with emulation software to the switch console port, or attach a PC to the Ethernet management port.
- The default data characteristics of the console port are 9600, 8, 1, no parity. You might need to press the Return key several times to see the command-line prompt.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**line vty 0 97** **Example:** `Device(config)# line vty 0 97`	Configures the number of Telnet sessions (lines), and enters line configuration mode. There are 98 possible sessions on a command-capable device. The 0 and 97 mean that you are configuring all 98 possible Telnet sessions.
Step 4	**password** {*unencrypted-password* | *encryption-type encrypted-password*} **Example:** `Device(config-line)# password 6 VeGSWf_NXfZOBPROFXVEaRf[TFeAAB`	Sets a Telnet password for the line or lines. For *encryption-type*, enter **0** to specify that an unencrypted password will follow. Enter **7** to specify that a hidden password will follow. Enter **6** to specify that an encrypted password will follow.
Step 5	**end** **Example:** `Device(config-line)# end`	Returns to privileged EXEC mode.

Configuring Username and Password Pairs

Follow these steps to configure username and password pairs:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**username** *name* [**privilege** *level*] {**password** *encryption-type password*} **Example:** `Device(config)# username adamsample privilege 1 password secret456` `Device(config)# username 111111111111 mac attribute`	Sets the username, privilege level, and password for each user. • For *name*, specify the user ID as one word or the MAC address. Spaces and quotation marks are not allowed. • You can configure a maximum of 12000 clients each, for both username and MAC filter. • (Optional) For *level*, specify the privilege level the user has after gaining access. The range is 0 to 15. Level 15 gives privileged EXEC mode access. Level 1 gives user EXEC mode access. • For *encryption-type*, enter **0** to specify that an unencrypted password will follow. Enter **7** to specify that a hidden password will follow. Enter **6** to specify that an encrypted password will follow. • For *password*, specify the password the user must enter to gain access to the device. The password must be from 1 to 25 characters, can contain embedded spaces, and must be the last option specified in the **username** command.
Step 4	Use one of the following: • **line console 0** • **line vty 0 97** **Example:**	Enters line configuration mode, and configures the console port (line 0) or the VTY lines (line 0 to 97).

	Command or Action	Purpose
	`Device(config)# line console 0` or `Device(config)# line vty 0 97`	
Step 5	**end** **Example:** `Device(config-line)# end`	Exits line configuration mode and returns to privileged EXEC mode.

Setting the Privilege Level for a Command

Follow these steps to set the privilege level for a command:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**privilege** *mode* **level** *level command* **Example:** `Device(config)# privilege exec level 14 configure`	Sets the privilege level for a command. • For *mode*, enter **configure** for global configuration mode, **exec** for EXEC mode, **interface** for interface configuration mode, or **line** for line configuration mode. • For *level*, the range is from 0 to 15. Level 1 is for normal user EXEC mode privileges. Level 15 is the level of access permitted by the **enable** password. • For *command*, specify the command to which you want to restrict access.
Step 4	**enable password level** *level password* **Example:** `Device(config)# enable password level 14 SecretPswd14`	Specifies the password to enable the privilege level. • For *level*, the range is from 0 to 15. Level 1 is for normal user EXEC mode privileges. • For *password*, specify a string from 1 to 25 alphanumeric characters. The string

	Command or Action	Purpose
		cannot start with a number, is case sensitive, and allows spaces but ignores leading spaces. By default, no password is defined.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Changing the Default Privilege Level for Lines

Follow these steps to change the default privilege level for the specified line:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**line vty** *line* **Example:** `Device(config)# line vty 10`	Selects the virtual terminal line on which to restrict access.
Step 4	**privilege exec level** *level* **Example:** `Device(config-line)# privilege exec level 15`	Changes the default privilege level for the line. For *level*, the range is from 0 to 15. Level 1 is for normal user EXEC mode privileges. Level 15 is the level of access permitted by the **enable** password.
Step 5	**end** **Example:** `Device(config-line)# end`	Exits line configuration mode and returns to privileged EXEC mode.

What to do next

Users can override the privilege level you set using the **privilege level** line configuration command by logging in to the line and enabling a different privilege level. They can lower the privilege level by using the **disable** command. If users know the password to a higher privilege level, they can use that password to enable the

higher privilege level. You might specify a high level or privilege level for your console line to restrict line usage.

Logging in to and Exiting a Privilege Level

Beginning in user EXEC mode, follow these steps to log into a specified privilege level and exit a specified privilege level.

Procedure

	Command or Action	Purpose
Step 1	**enable** *level* **Example:** `Device> enable 15`	Logs in to a specified privilege level. InI the example, Level 15 is privileged EXEC mode. For *level*, the range is 0 to 15.
Step 2	**disable** *level* **Example:** `Device# disable 1`	Exits to a specified privilege level. In the example, Level 1 is user EXEC mode. For *level*, the range is 0 to 15.

Configuring an Encrypted Preshared Key

To configure an encrypted preshared key, perform the following steps.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**key config-key password-encrypt** [*text*] **Example:** `Device(config)# key config-key password-encrypt`	Stores a type 6 encryption key in private NVRAM. • To key in interactively (using the **Enter** key) and an encrypted key already exists, you will be prompted for the following: Old key, New key, and Confirm key. • To key in interactively, but an encryption key is not present, you will be prompted

	Command or Action	Purpose
		for the following: New key and Confirm key. • When removing the password that is already encrypted, you will see the following prompt: `WARNING: All type 6 encrypted keys will become unusable. Continue with master key deletion? [yes/no]:"`
Step 4	**password encryption aes** **Example:** `Device(config)# password encryption aes`	Enables the encrypted preshared key.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Monitoring Switch Access with Passwords and Privileges

Table 86: Commands for Displaying Privilege-Level Information

Command	Information
show privilege	Displays the privilege level configuration.

Configuration Examples for Switch Access with Passwords and Privilege Levels

Example: Setting or Changing a Static Enable Password

The following example shows how to change the enable password to *l1u2c3k4y5*. The password is not encrypted and provides access to level 15 (traditional privileged EXEC mode access):

```
Device> enable
Device# configure terminal
Device(config)# enable password l1u2c3k4y5
Device(config)# end
```

Example: Protecting Enable and Enable Secret Passwords with Encryption

The following example shows how to configure the encrypted password *9sMLBsTFXLnnHTk$0L82* for privilege level 2:

```
Device> enable
Device# configure terminal
Device(config)# enable secret level 2 9 $9$sMLBsTFXLnnHTk$0L82
Device(config)# end
```

Example: Setting a Telnet Password for a Terminal Line

The following example shows how to set the Telnet password to *let45me67in89*:

```
Device> enable
Device# configure terminal
Device(config)# line vty 10
Device(config-line)# password let45me67in89
Device(config-line)# end
```

Example: Setting the Privilege Level for a Command

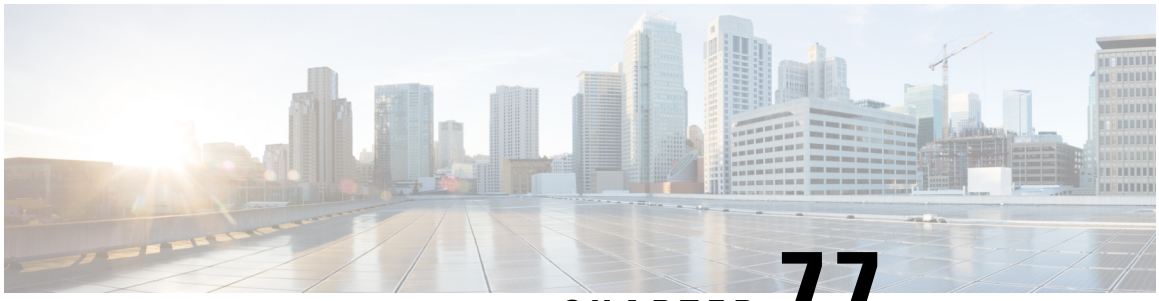
The following example shows how to set the **configure** command to privilege level 14 and define *SecretPswd14* as the password users must enter to use level 14 commands:

```
Device> enable
Device# configure terminal
Device(config)# privilege exec level 14 configure
Device(config)# enable password level 14 SecretPswd14
Device(config)# end
```

Example: Configuring an Encrypted Preshared Key

The following example shows a configuration for which a type 6 preshared key has been encrypted. It includes the prompts and messages that a user might see.

```
Device> enable
Device# configure terminal
Device(config)# password encryption aes
Device(config)# key config-key password-encrypt
New key:
Confirm key:
Device(config)#
01:46:40: TYPE6_PASS: New Master key configured, encrypting the keys with
the new master key
Device(config)# end
```



CHAPTER 77

Configuring Authentication

Authentication provides a method to identify users, which includes the login and password dialog, challenge and response, messaging support, and encryption, depending on the selected security protocol. Authentication is the way a user is identified prior to being allowed access to the network and network services.

Prerequisites for Configuring Authentication

The implementation of authentication is divided into Authentication, Authorization, and Accounting (AAA) authentication and nonauthentication methods. Cisco recommends that, whenever possible, AAA security services be used to implement authentication.

Restrictions for Configuring Authentication

- The number of AAA method lists that can be configured is 250.
- If you configure the same RADIUS server IP address for a different UDP destination port for accounting requests by using the **acct-port** keyword and a UDP destination port for authentication requests by using the **auth-port** keyword with and without the nonstandard option, the RADIUS server does not accept the nonstandard option.

Information About Authentication

Named Method Lists for Authentication

A named list of authentication methods is first defined before AAA authentication can be configured, and the named list is then applied to various interfaces. The method list defines the types of authentication and the sequence in which they are performed; it must be applied to a specific interface before any of the defined authentication methods are performed. The only exception is the default method list (which is named “default”).

The default method list is automatically applied to all interfaces, except those that have a named method list explicitly defined. A defined method list overrides the default method list.

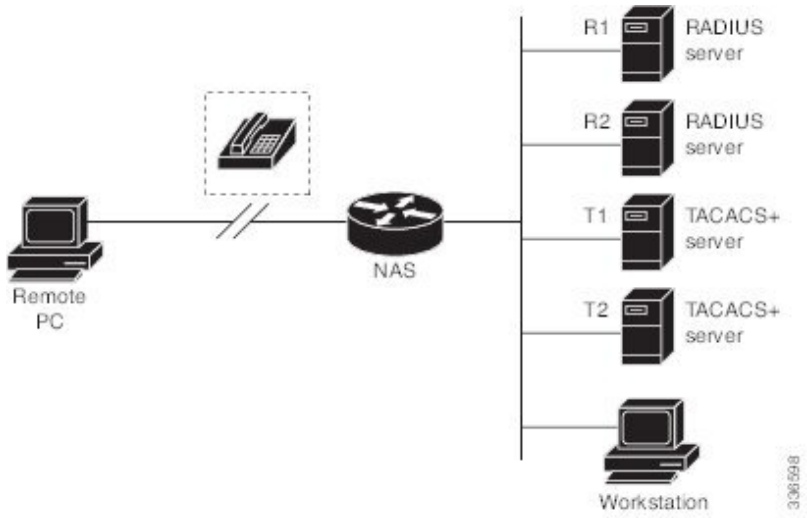
A method list is a sequential list describing the authentication methods to be queried to authenticate a user. Method lists enable you to designate one or more security protocols to be used for authentication, thus ensuring a backup system for authentication in case the initial method fails. Cisco software uses the first listed method to authenticate users. If that method fails to respond, the Cisco software selects the next authentication method listed in the method list. This process continues until there is successful communication with a listed authentication method, or all methods defined in the method list are exhausted.

Note that the software attempts authentication with the next listed authentication method only when there is no response from the previous method. If authentication fails at any point in this cycle, that is, the security server or local username database responds by denying the user access, then the authentication process stops and no other authentication methods are attempted.

Method Lists and Server Groups

A server group is a way to group existing RADIUS or TACACS+ server hosts for use in method lists. The figure below shows a typical AAA network configuration that includes four security servers: R1 and R2 are RADIUS servers and T1 and T2 are TACACS+ servers. R1 and R2 make up the group of RADIUS servers. T1 and T2 make up the group of TACACS+ servers.

Figure 85: Configuration of a Typical AAA Network



Using server groups, you can specify a subset of the configured server hosts and use them for a particular service. For example, server groups allow you to define R1 and R2 as a server group, and define T1 and T2 as a separate server group.

Server groups also can include multiple host entries for the same server, as long as each entry has a unique identifier. The combination of an IP address and a UDP port number creates a unique identifier, allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. In other words, this unique identifier enables RADIUS requests to be sent to different UDP ports on a server at the same IP address. If two different host entries on the same RADIUS server are configured for the same service--for example, authentication--the second host entry configured acts as failover backup to the first one. Using this example, if the first host entry fails to provide accounting services, the network access server will try the second host entry configured on the same device for accounting services. (The RADIUS host entries will be tried in the order in which they are configured.)

For more information about configuring server groups and about configuring server groups based on Dialed Number Identification Service (DNIS) numbers, see the "Configuring RADIUS" or "Configuring TACACS+" chapters.

Login Authentication Using AAA

Login Authentication Using Enable Password

Use the **aaa authentication login**command with the **enable** keyword to specify the enable password as the login authentication method. For example, to specify the enable password as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default enable
```

Before you can use the enable password as the login authentication method, you need to define the enable password. For more information about defining enable passwords, see chapter "Controlling Switch Access with Passwords and Privilege Levels."

Login Authentication Using Line Password

Use the **aaa authentication login default** command with the **line** keyword to specify the line password as the login authentication method. For example, to specify the line password as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default line
```

Before you can use a line password as the login authentication method, you need to define a line password.

Login Authentication Using Local Password

Use the **aaa authentication login default**command with the **local** keyword to specify that the Cisco device will use the local username database for authentication. For example, to specify the local username database as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default local
```

Login Authentication Using Group RADIUS

Use the **aaa authentication login default**command with the **group radius** to specify RADIUS as the login authentication method. For example, to specify RADIUS as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default group radius
```

Before you can use RADIUS as the login authentication method, you need to enable communication with the RADIUS security server. For more information about establishing communication with a RADIUS server, refer to the chapter "Configuring RADIUS."

RADIUS Attribute 8 in Access Requests

After you have used the **aaa authentication login** command to specify RADIUS and your login host has been configured to request its IP address from the NAS, you can send attribute 8 (Framed-IP-Address) in access-request packets by using the **radius-server attribute 8 include-in-access-req** command in global configuration mode. This command makes it possible for NAS to provide the RADIUS server a hint of the user IP address in advance for user authentication.

Login Authentication Using Group TACACS

Use the **aaa authentication login default** command with the **group tacacs+** to specify TACACS+ as the login authentication method. For example, to specify TACACS+ as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default group tacacs+
```

Before you can use TACACS+ as the login authentication method, you need to enable communication with the TACACS+ security server. For more information about establishing communication with a TACACS+ server, refer to the chapter "Configuring TACACS+."

Login Authentication Using Group Name

Use the **aaa authentication login default** command with the **group** *group-name* method to specify a subset of RADIUS or TACACS+ servers to use as the login authentication method. To specify and define the group name and the members of the group, use the **aaa group server** command. For example, use the **aaa group server** command to first define the members of **group loginrad**:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius loginrad
Device(config-sg-radius)# server 172.16.2.3
Device(config-sg-radius)# server 172.16.2 17
Device(config-sg-radius)# server 172.16.2.32
Device(config-sg-radius)# end
```

This command specifies RADIUS servers 172.16.2.3, 172.16.2.17, and 172.16.2.32 as members of the group *loginrad*.

To specify **group loginrad** as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default group loginrad
```

Before you can use a group name as the login authentication method, you need to enable communication with the RADIUS or TACACS+ security server. For more information about establishing communication with a RADIUS server, refer to the chapter "Configuring RADIUS." For more information about establishing communication with a TACACS+ server, refer to the chapter "Configuring TACACS+."

Specifying the Amount of Time for Login Input

The **timeout login response**command allows you to specify how long the system will wait for login input (such as username and password) before timing out. The default login value is 30 seconds; with the **timeout login response** command, you can specify a timeout value from 1 to 300 seconds. To change the login timeout value from the default of 30 seconds, use the following command in line configuration mode:

```
Device(config-line)# timeout login response 30
```

Password Protection at the Privileged Level

Use the **aaa authentication enable default** command to create a series of authentication methods that are used to determine whether a user can access the privileged EXEC command level. You can specify up to four authentication methods. The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

Use the following command in global configuration mode:

```
Device(config)# authentication enable default radius
```

or

```
Device(config)# authentication enable default tacacs
```

Changing the Text Displayed at the Password Prompt

Use the **aaa authentication password-prompt** command to change the default text that the Cisco IOS XE software displays when prompting a user to enter a password. This command changes the password prompt for the enable password as well as for login passwords that are not supplied by remote security servers. The **no** form of this command returns the password prompt to the following default value:

```
Password:
```

The **aaa authentication password-prompt** command does not change any dialog that is supplied by a remote TACACS+ or RADIUS server.

The **aaa authentication password-prompt** command works when RADIUS is used as the login method. You will be able to see the password prompt defined in the command shown even when the RADIUS server is unreachable. The **aaa authentication password-prompt** command does not work with TACACS+. TACACS+ supplies the NAS with the password prompt to display to the users. If the TACACS+ server is reachable, the NAS gets the password prompt from the server and uses that prompt instead of the one defined in the **aaa authentication password-prompt** command. If the TACACS+ server is not reachable, the password prompt defined in the **aaa authentication password-prompt** command may be used.

Use the following command in global configuration mode:

```
Device(config)# aaa authentication password-prompt "Enter your password now:"
```

Domain Stripping

The AAA Broadcast Accounting feature allows accounting information to be sent to multiple AAA servers at the same time, that is, accounting information can be broadcast to one or more AAA servers simultaneously. This functionality allows you to send accounting information to private and public AAA servers. It also provides redundant billing information for voice applications.

The Domain Stripping feature allows domain stripping to be configured at the server group level.

Per-server group configuration overrides the global configuration. If domain stripping is not enabled globally, but it is enabled in a server group, then it is enabled only for that server group. Also, if virtual routing and forwarding (VRF)-specific domain stripping is configured globally and in a server group for a different VRF, domain stripping is enabled in both the VRFs. VRF configurations are taken from server-group configuration mode. If server-group configurations are disabled in global configuration mode but are available in server-group configuration mode, all configurations in server-group configuration mode are applicable.

After the domain stripping and broadcast accounting are configured, you can create separate accounting records as per the configurations.

If both **domain-stripping** and **directed-request** commands are enabled, domain stripping takes precedence and directed request functionality will not work.

How to Configure Authentication

Configuring Login Authentication Using AAA

The AAA security services facilitate a variety of login authentication methods. Use the **aaa authentication login**command to enable AAA authentication no matter which of the supported login authentication methods you decide to use. With the **aaa authentication login**command, you create one or more lists of authentication methods that are tried at login. These lists are applied using the **login authentication** line configuration command.

To configure login authentication by using AAA, use the following commands beginning in global configuration mode:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**aaa authentication login** {**default** | *list-name*} *method1*[*method2...*] **Example:** `Device(config)# aaa authentication login default local`	Creates a local authentication list.
Step 5	**line** [**aux** | **console** | **tty** | **vty**] **line-number** [**ending-line-number**] **Example:** `Device(config)# line vty 1`	Enters line configuration mode for the lines to which you want to apply the authentication list.
Step 6	**login authentication** {**default** | *list-name*} **Example:** `Device(config-line)# login authentication default`	Applies the authentication list to a line or set of lines.

	Command or Action	Purpose
Step 7	**end** **Example:** `Device(config-line)# end`	Exits line configuration mode and returns to privileged EXEC mode.

What to do next

The *list-name* is a character string used to name the list you are creating. The method argument refers to the actual method the authentication algorithm tries. The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

For example, to specify that authentication should succeed even if (in this example) the TACACS+ server returns an error, enter the following command:

```
Device(config)# aaa authentication login default group tacacs+ none
```

Note Because the **none** keyword enables *any* user logging in to successfully authenticate, it should be used only as a backup method of authentication.

To create a default list that is used when a named list is *not* specified in the **login authentication** command, use the **default** keyword followed by the methods that are to be used in default situations. The default method list is automatically applied to all interfaces.

For example, to specify RADIUS as the default method for user authentication during login, enter the following command:

```
Device(config)# aaa authentication login default group radius
```

Preventing an Access Request with a Blank Username from Being Sent to the RADIUS Server

The following configuration steps provide the ability to prevent an Access Request with a blank username from being sent to the RADIUS server. This functionality ensures that unnecessary RADIUS server interaction is avoided, and RADIUS logs are kept short.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA globally.
Step 4	**aaa authentication suppress null-username** **Example:** `Device(config)# aaa authentication suppress null-username`	Prevents an Access Request with a blank username from being sent to the RADIUS server.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Message Banners for AAA Authentication

AAA supports the use of configurable, personalized login and failed-login banners. You can configure message banners that will be displayed when a user logs in to the system to be authenticated using AAA and when, for whatever reason, authentication fails.

Configuring a Login Banner

To configure a banner that is displayed when a user logs in (replacing the default message for login), perform the following task:

Before you begin

To create a login banner, you must configure a delimiting character that notifies the system that the following text string must be displayed as the banner, and then the text string itself. The delimiting character is repeated at the end of the text string to signify the end of the banner. The delimiting character can be any single character in the extended ASCII character set, but once defined as the delimiter, that character cannot be used in the text string for the banner.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:**	Enables AAA.

	Command or Action	Purpose
	`Device(config)# aaa new-model`	
Step 4	**aaa authentication banner** *delimiter string delimiter* **Example:** `Device(config)# aaa authentication banner *Unauthorized use is prohibited.*`	Creates a personalized login banner.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring a Failed-Login Banner

To configure a message that is displayed when a user login fails (replacing the default message for failed login), perform the following task:

Before you begin

To create a failed-login banner, you must configure a delimiting character, which notifies the system that the following text string must be displayed as the banner, and then configure the text string itself. The delimiting character is repeated at the end of the text string to signify the end of the failed-login banner. The delimiting character can be any single character in the extended ASCII character set, but once defined as the delimiter, that character cannot be used in the text string making up the banner.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**aaa authentication fail-message** *delimiter string delimiter* **Example:** `Device(config)# aaa authentication fail-message *Failed login. Try again.*`	Creates a message to be displayed when a user login fails.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring AAA Packet of Disconnect

Packet of disconnect (POD) terminates connections on the network access server (NAS) when particular session attributes are identified. By using session information obtained from AAA, the POD client residing on a UNIX workstation sends disconnect packets to the POD server running on the network access server. The NAS terminates any inbound user session with one or more matching key attributes. It rejects requests when required fields are missing or when an exact match is not found.

To configure POD, perform the following tasks in global configuration mode:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa accounting network default start-stop radius** **Example:** `Device(config)# aaa accounting network default start-stop radius`	Enables AAA accounting records.
Step 4	**aaa accounting delay-start** **Example:** `Device(config)# aaa accounting delay-start`	(Optional) Delays generation of the start accounting record until the Framed-IP-Address is assigned, allowing its use in the POD packet.
Step 5	**aaa pod server server-key** *string* **Example:** `Device(config)# aaa pod server server-key xyz123`	Enables POD reception.
Step 6	**radius server** *name* non-standard **Example:** `Device(config)# radius server radser`	Configures a RADIUS server and enters RADIUS server configuration mode.

	Command or Action	Purpose
Step 7	**address** {**ipv4** | **ipv6**} *hostname* **Example:** `Device(config-radius-server)# address ipv4 radius-host`	Configures a RADIUS host.
Step 8	**end** **Example:** `Device(config-radius-server)# end`	Exits RADIUS server configuration mode and returns to privileged EXEC mode.

Configuring Domain Stripping at the Server Group Level

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa group server radius** *server-name* **Example:** `Device(config)# aaa group server radius rad1`	Adds the RADIUS server and enters server group RADIUS configuration mode. • The *server-name* argument specifies the RADIUS server group name.
Step 4	**domain-stripping** [**strip-suffix** *word*] [**right-to-left**] [**prefix-delimiter** *word*] [**delimiter** *word*] **Example:** `Device(config-sg-radius)# domain-stripping delimiter username@example.com`	Configures domain stripping at the server group level.
Step 5	**end** **Example:** `Device(config-sg-radius)# end`	Exits server group RADIUS configuration mode and returns to the privileged EXEC mode.

Configuring Non-AAA Authentication Methods

Configuring Line Password Protection

This task is used to provide access control on a terminal line by entering the password and establishing password checking.

Note If you configure line password protection and then configure TACACS or extended TACACS, the TACACS username and password take precedence over line passwords. If you have not yet implemented a security policy, we recommend that you use AAA.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**line** [**aux** | **console** | **tty** | **vty**] *line-number* [*ending-line-number*] **Example:** `Device(config)# line console 0`	Enters line configuration mode.
Step 4	**password** *password* **Example:** `Device(config-line)# secret word`	Assigns a password to a terminal or other device on a line. The password checker is case sensitive and can include spaces; for example, the password "Secret" is different from the password "secret," and "two words" is an acceptable password.
Step 5	**login** **Example:** `Device(config-line)# login`	Enables password checking at login. You can disable line password verification by disabling password checking by using the **no** version of this command. **Note** The **login** command only changes username and privilege level but it does not execute a shell; therefore autocommands will not be executed. To execute autocommands under this circumstance, you need to establish a Telnet session back into the device (loop-back). Make

	Command or Action	Purpose
		sure that the device has been configured for secure Telnet sessions if you choose to implement autocommands this way.
Step 6	**end** **Example:** `Device(config-line)# end`	Exits line configuration mode and returns to privileged EXEC mode.

Establishing Username Authentication

You can create a username-based authentication system, which is useful in the following situations:

- To provide a TACACS-like username and encrypted password-authentication system for networks that cannot support TACACS
- To provide special-case logins: for example, access list verification, no password verification, autocommand execution at login, and "no escape" situations

To establish username authentication, perform the following task:

Procedure

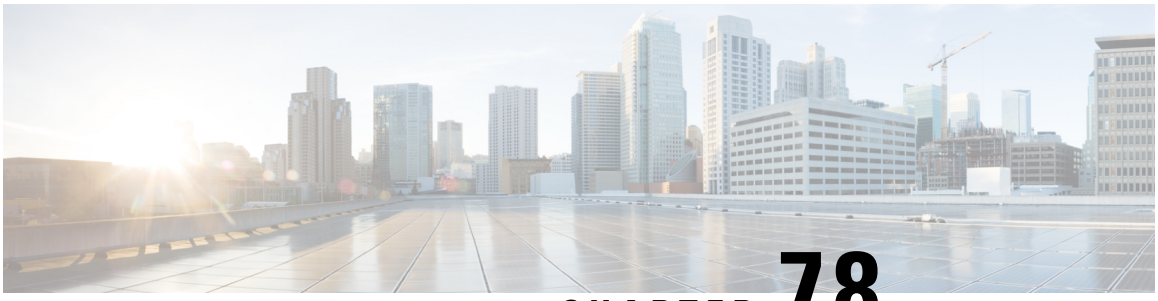
	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	Do one of the following: • **username** *name* [**nopassword** | **password** *password* | **password** *encryption-type encrypted password*] • **username** *name* [**access-class** *number*] **Example:** `Device(config)# username superuser password superpassword password 7 encrypted-password` `Device(config)# username user1 access-class access-user`	Establishes username authentication with encrypted passwords. or (Optional) Establishes username authentication by access list.
Step 4	**username** *name* [**privilege** *level*] **Example:**	(Optional) Sets the privilege level for the user.

	Command or Action	Purpose
	`Device(config)# username user1 privilege 5`	
Step 5	**username** *name* [**autocommand** *command*] **Example:** `Device(config)# username user1 autocommand show users`	(Optional) Specifies a command to be executed automatically.
Step 6	**username** *name* [**noescape**] [**nohangup**] **Example:** `Device(config)# username user1 noescapee`	(Optional) Sets a "no escape" login environment.
Step 7	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

What to do next

The keyword **noescape** prevents users from using escape characters on the hosts to which they are connected. The **nohangup** feature does not disconnect after using the autocommand.

Caution Passwords will be displayed in clear text in your configuration unless you enable the **service password-encryption** command.



CHAPTER 78

Configuring Authorization

AAA authorization enables you to limit the services available to a user. When AAA authorization is enabled, the network access server uses information retrieved from the user's profile, which is located either in the local user database or on the security server, to configure the user's session. Once this is done, the user will be granted access to a requested service only if the information in the user profile allows it.

Prerequisites for Configuring Authorization

Before configuring authorization using named method lists, you must first perform the following tasks:

- Enable authentication, authorization, and accounting (AAA) on your network access server.
- Configure AAA authentication. Authorization generally takes place after authentication and relies on authentication to work properly. For more information about AAA authentication, refer to the "Configuring Authentication" module.
- Define the characteristics of your RADIUS or TACACS+ security server if you are issuing RADIUS or TACACS+ authorization. For more information about configuring your Cisco network access server to communicate with your RADIUS security server, refer to the chapter "Configuring RADIUS". For more information about configuring your Cisco network access server to communicate with your TACACS+ security server, refer to the "Configuring TACACS+" module.
- Define the rights associated with specific users by using the **username** command if you are issuing local authorization.

Information About Configuring Authorization

Named Method Lists for Authorization

Method lists for authorization define the ways that authorization will be performed and the sequence in which these methods will be performed. A method list is simply a named list describing the authorization methods

to be queried (such as RADIUS or TACACS+), in sequence. Method lists enable you to designate one or more security protocols to be used for authorization, thus ensuring a backup system in case the initial method fails. Cisco IOS XE software uses the first method listed to authorize users for specific network services; if that method fails to respond, the Cisco IOS XE software selects the next method listed in the list. This process continues until there is successful communication with a listed authorization method, or all methods defined are exhausted.

Note The Cisco IOS XE software attempts authorization with the next listed method only when there is no response from the previous method. If authorization fails at any point in this cycle--meaning that the security server or local username database responds by denying the user services--the authorization process stops and no other authorization methods are attempted.

Method lists are specific to the authorization type requested:

- Commands: Applies to the EXEC mode commands a user issues. Command authorization attempts authorization for all EXEC mode commands, including global configuration commands, associated with a specific privilege level.
- EXEC: Applies to the attributes associated with a user EXEC terminal session.
- Network: Applies to network connections. This can include a SLIP or ARAP connection.
- Reverse Access: Applies to reverse Telnet sessions.

When you create a named method list, you are defining a particular list of authorization methods for the indicated authorization type.

Once defined, method lists must be applied to specific lines or interfaces before any of the defined methods will be performed. The only exception is the default method list (which is named "default"). If the **aaa authorization** command for a particular authorization type is issued without a named method list specified, the default method list is automatically applied to all interfaces or lines except those that have a named method list explicitly defined. (A defined method list overrides the default method list.) If no default method list is defined, local authorization takes place by default.

AAA Authorization Methods

AAA supports five different methods of authorization:

- TACACS+: The network access server exchanges authorization information with the TACACS+ security daemon. TACACS+ authorization defines specific rights for users by associating attribute-value pairs, which are stored in a database on the TACACS+ security server, with the appropriate user.
- If-Authenticated: The user is allowed to access the requested function provided the user has been authenticated successfully.
- None: The network access server does not request authorization information; authorization is not performed over this line/interface.
- Local: The router or access server consults its local database, as defined by the **username** command, for example, to authorize specific rights for users. Only a limited set of functions can be controlled via the local database.

- RADIUS: The network access server requests authorization information from the RADIUS security server. RADIUS authorization defines specific rights for users by associating attributes, which are stored in a database on the RADIUS server, with the appropriate user.

Note With CSCuc32663, passwords and authorization logs are masked before being sent to the TACACS+, LDAP, or RADIUS security servers. Use the **aaa authorization commands visible-keys** command to send unmasked information to the TACACS+, LDAP, or RADIUS security servers.

Authorization Methods

To have the network access server request authorization information via a TACACS+ security server, use the **aaa authorization** command with the **group tacacs+** *method* keyword. For more specific information about configuring authorization using a TACACS+ security server, refer to the chapter "Configuring TACACS+." For an example of how to enable a TACACS+ server to authorize the use of network services, see the TACACS Authorization Examples.

To allow users to have access to the functions they request as long as they have been authenticated, use the **aaa authorization** command with the **if-authenticated** *method* keyword. If you select this method, all requested functions are automatically granted to authenticated users.

There may be times when you do not want to run authorization from a particular interface or line. To stop authorization activities on designated lines or interfaces, use the **none** *method* keyword. If you select this method, authorization is disabled for all actions.

To select local authorization, which means that the router or access server consults its local user database to determine the functions a user is permitted to use, use the **aaa authorization** command with the **local** *method* keyword. The functions associated with local authorization are defined by using the **username** global configuration command. For a list of permitted functions, refer to the chapter "Configuring Authentication."

To have the network access server request authorization via a RADIUS security server, use the **radius** *method* keyword. For more specific information about configuring authorization using a RADIUS security server, refer to the Configuring RADIUS chapter.

To have the network access server request authorization via a RADIUS security server, use the **aaa authorization** command with the **group radius** *method* keyword. For more specific information about configuring authorization using a RADIUS security server, refer to the chapter Configuring RADIUS. For an example of how to enable a RADIUS server to authorize services, see the RADIUS Authorization Example.

Method Lists and Server Groups

A server group is a way to group existing RADIUS or TACACS+ server hosts for use in method lists. The figure below shows a typical AAA network configuration that includes four security servers: R1 and R2 are RADIUS servers, and T1 and T2 are TACACS+ servers. R1 and R2 make up the group of RADIUS servers. T1 and T2 make up the group of TACACS+ servers.

Using server groups, you can specify a subset of the configured server hosts and use them for a particular service. For example, server groups allow you to define R1 and R2 as separate server groups, and T1 and T2 as separate server groups. This means you can specify either R1 and T1 in the method list or R2 and T2 in the method list, which provides more flexibility in the way that you assign RADIUS and TACACS+ resources.

Server groups also can include multiple host entries for the same server, as long as each entry has a unique identifier. The combination of an IP address and a UDP port number creates a unique identifier, allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. In other words, this unique identifier enables RADIUS requests to be sent to different UDP ports on a server at the same IP address. If two different host entries on the same RADIUS server are configured for the same service--for example, authorization--the second host entry configured acts as fail-over backup to the first one. Using this example, if the first host entry fails to provide accounting services, the network access server will try the second host entry configured on the same device for accounting services. (The RADIUS host entries will be tried in the order they are configured.)

For more information about configuring server groups and about configuring server groups based on DNIS numbers, refer to the chapter Configuring RADIUS or the chapter Configuring TACACS+.

AAA Authorization Types

Cisco IOS XE software supports five different types of authorization:

- Commands: Applies to the EXEC mode commands a user issues. Command authorization attempts authorization for all EXEC mode commands, including global configuration commands, associated with a specific privilege level.
- EXEC: Applies to the attributes associated with a user EXEC terminal session.
- Network: Applies to network connections. This can include a SLIP or ARAP connection.
- Reverse Access: Applies to reverse Telnet sessions.
- Configuration: Applies to downloading configurations from the AAA server.
- IP Mobile: Applies to authorization for IP mobile services.

Authorization Types

Named authorization method lists are specific to the indicated type of authorization.

To create a method list to enable authorization that applies specific security policies on a per-user basis, use the auth-proxy keyword. For detailed information on the authentication proxy feature, refer to the chapter “Configuring Authentication Proxy” in the “Traffic Filtering and Firewalls” part of this book.

To create a method list to enable authorization for all network-related service requests (including SLIP and ARAP), use the **network** keyword.

To create a method list to enable authorization to determine if a user is allowed to run an EXEC shell, use the **exec** keyword.

To create a method list to enable authorization for specific, individual EXEC commands associated with a specific privilege level, use the **commands**keyword. (This allows you to authorize all commands associated with a specified command level from 0 to 15.)

To create a method list to enable authorization for reverse Telnet functions, use the **reverse-access** keyword.

For information about the types of authorization supported by the Cisco IOS XE software, refer to the AAA Authorization Types.

Authorization Attribute-Value Pairs

RADIUS and TACACS+ authorization both define specific rights for users by processing attributes, which are stored in a database on the security server. For both RADIUS and TACACS+, attributes are defined on the security server, associated with the user, and sent to the network access server where they are applied to the user's connection.

For a list of supported RADIUS attributes, refer to the "RADIUS Attributes Overview and RADIUS IETF Attributes" chapter. For a list of supported TACACS+ AV pairs, refer to the "Configuring TACACS+" chapter.

How to Configure Authorization

Disabling Authorization for Global Configuration Commands

The **aaa authorization** command with the keyword **commands** attempts authorization for all EXEC mode commands, including global configuration commands, associated with a specific privilege level. Because there are configuration commands that are identical to some EXEC-level commands, there can be some confusion in the authorization process. Using **no aaa authorization config-commands** stops the network access server from attempting configuration command authorization.

When the **no aaa authorization console** command is configured along with the **aaa authentication enable default enable** command, enable mode will be locked unless enable password or enable secret is explicitly configured on the device.

To disable AAA authorization for all global configuration commands, use the following command in global configuration mode:

Command	Purpose
`Device(config)# no aaa authorization config-commands`	Disables authorization for all global configuration commands.

To disable AAA authorization on the console, use the following command in global configuration mode:

Note AAA authorization is disabled on the console by default. If AAA authorization is enabled on the console, disable it by configuring the **no aaa authorization console** command during the AAA configuration stage. AAA should be disabled on the console for user authentication.

Command	Purpose
`Device(config)# no aaa authorization console`	Disables authorization on the console.

Configuring Authorization for Reverse Telnet

Telnet is a standard terminal emulation protocol used for remote terminal connection. Normally, you log in to a network access server (typically through a dialup connection) and then use Telnet to access other network devices from that network access server. There are times, however, when it is necessary to establish a reverse Telnet session. In reverse Telnet sessions, the Telnet connection is established in the opposite direction--from inside a network to a network access server on the network periphery to gain access to modems or other devices connected to that network access server. Reverse Telnet is used to provide users with dialout capability by allowing them to Telnet to modem ports attached to a network access server.

It is important to control access to ports accessible through reverse Telnet. Failure to do so could, for example, allow unauthorized users free access to modems where they can trap and divert incoming calls or make outgoing calls to unauthorized destinations.

Authentication during reverse Telnet is performed through the standard AAA login procedure for Telnet. Typically the user has to provide a username and password to establish either a Telnet or reverse Telnet session. Reverse Telnet authorization provides an additional (optional) level of security by requiring authorization in addition to authentication. When enabled, reverse Telnet authorization can use RADIUS or TACACS+ to authorize whether or not this user is allowed reverse Telnet access to specific asynchronous ports, after the user successfully authenticates through the standard Telnet login procedure.

Reverse Telnet authorization offers the following benefits:

- An additional level of protection by ensuring that users engaged in reverse Telnet activities are indeed authorized to access a specific asynchronous port using reverse Telnet.
- An alternative method (other than access lists) to manage reverse Telnet authorization.

To configure a network access server to request authorization information from a TACACS+ or RADIUS server before allowing a user to establish a reverse Telnet session, use the following command in global configuration mode:

Command	Purpose
`Device(config)# aaa authorization reverse-access` *method1* [*method2* ...]	Configures the network access server to request authorization information before allowing a user to establish a reverse Telnet session.

This feature enables the network access server to request reverse Telnet authorization information from the security server, whether RADIUS or TACACS+. You must configure the specific reverse Telnet privileges for the user on the security server itself.

Configuration Examples for Authorization

Example: TACACS Authorization

The following examples show how to use a TACACS+ server to authorize the use of network services, including ARA. If the TACACS+ server is not available or an error occurs during the authorization process, the fallback method (none) is to grant all authorization requests:

```
Device(config)# aaa authorization network default group tacacs+ none
```

The following example shows how to allow network authorization using TACACS+:

```
Device(config)# aaa authorization network default group tacacs+
```

The following example shows how to provide the same authorization, but it also creates address pools called "*mci*" and "*att*":

```
Device> enable
Device# configure terminal
Device(config)# aaa authorization network default group tacacs+
Device(config)# interface gigabitethernet 1/1
Device(config-if)# ip address-pool local
Device(config-if)# exit
Device(config)# ip local-pool mci 172.16.0.1 172.16.0.255
Device(config)# ip local-pool att 172.17.0.1 172.17.0.255
Device(config-if)# end
```

Example: RADIUS Authorization

The following example shows how to configure the router to authorize using RADIUS:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authorization exec default group radius if-authenticated
Device(config)# aaa authorization network default group radius
Device(config)# radius server ip
Device(config-radius-server)# key sharedkey
Device(config-radius-server)# end
```

The lines in this sample RADIUS authorization configuration are defined as follows:

- The **aaa authorization exec default group radius if-authenticated** command configures the network access server to contact the RADIUS server to determine if users are permitted to start an EXEC shell when they log in. If an error occurs when the network access server contacts the RADIUS server, the fallback method is to permit the CLI to start, provided the user has been properly authenticated.

The RADIUS information returned may be used to specify an autocommand or a connection access list be applied to this connection.

- The **aaa authorization network default group radius** command configures network authorization via RADIUS. This can be used to govern address assignment, the application of access lists, and various other per-user quantities.

Note Because no fallback method is specified in this example, authorization will fail if, for any reason, there is no response from the RADIUS server.

Example: Reverse Telnet Authorization

The following examples show how to cause the network access server to request authorization information from a TACACS+ security server before allowing a user to establish a reverse Telnet session:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication login default group tacacs+
Device(config)# aaa authorization reverse-access default group tacacs+
Device(config)# tacacs server server1
Device(config-server-tacacs)# address ipv4 172.31.255.0
Device(config-server-tacacs)# timeout 90
Device(config-server-tacacs)# key sharedkey
Device(config-server-tacacs)# end
```

The lines in this sample TACACS+ reverse Telnet authorization configuration are defined as follows:

- The **aaa new-model** command enables AAA.
- The **aaa authentication login default group tacacs+** command specifies TACACS+ as the default method for user authentication during login.
- The **aaa authorization reverse-access default group tacacs+** command specifies TACACS+ as the method for user authorization when trying to establish a reverse Telnet session.
- The **tacacs server** command identifies the TACACS+ server.
- The **timeout** command sets the interval of time that the network access server waits for the TACACS+ server to reply.
- The **key** command defines the encryption key used for all TACACS+ communications between the network access server and the TACACS+ daemon.

The following example shows how to configure a generic TACACS+ server to grant a user, pat, reverse Telnet access to port tty2 on the network access server named “maple” and to port tty5 on the network access server named “oak”:

```
user = pat
  login = cleartext lab
  service = raccess {
    port#1 = maple/tty2
    port#2 = oak/tty5
```

Note In this example, “maple” and “oak” are the configured host names of network access servers, not DNS names or alias.

The following example shows how to configure the TACACS+ server (CiscoSecure) to grant a user named pat reverse Telnet access:

```
user = pat
profile_id = 90
profile_cycle = 1
member = Tacacs_Users
service=shell {
default cmd=permit
}
service=raccess {
allow “c2511e0” “tty1” “.*”
refuse “.*” “.*” “.*”
password = clear “goaway”
```

Note CiscoSecure only supports reverse Telnet using the command line interface in versions 2.1(*x*) through version 2.2(1).

An empty “service=raccess {}” clause permits a user to have unconditional access to network access server ports for reverse Telnet. If no “service=raccess” clause exists, the user is denied access to any port for reverse Telnet.

For more information about configuring TACACS+, refer to the “Configuring TACACS” chapter. For more information about configuring CiscoSecure, refer to the *CiscoSecure Access Control Server User Guide* , version 2.1(2) or greater.

The following example shows how to cause the network access server to request authorization from a RADIUS security server before allowing a user to establish a reverse Telnet session:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication login default group radius
Device(config)# aaa authorization reverse-access default group radius
Device(config)# radius server ip
Device(config-radius-server)# key sharedkey
Device(config-radius-server)# address ipv4 172.31.255.0 auth-port 1645 acct-port 1646
Device(config-radius-server)# end
```

The lines in this sample RADIUS reverse Telnet authorization configuration are defined as follows:

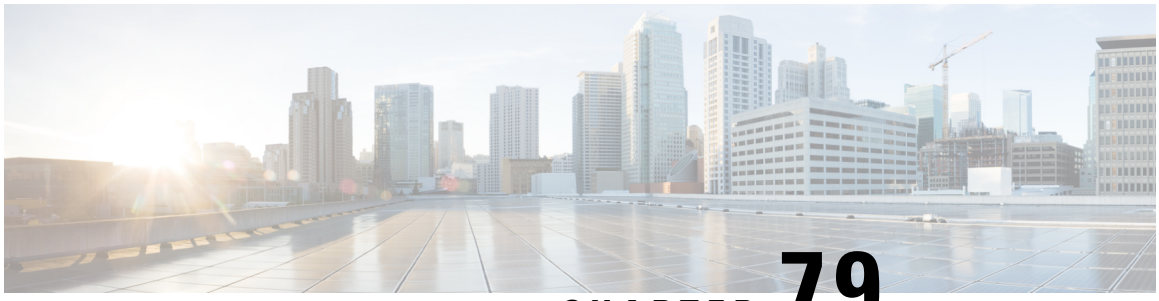
- The **aaa new-model** command enables AAA.
- The **aaa authentication login default group radius** command specifies RADIUS as the default method for user authentication during login.
- The **aaa authorization reverse-access default group radius** command specifies RADIUS as the method for user authorization when trying to establish a reverse Telnet session.
- The **radius** command identifies the RADIUS server.
- The **key** command defines the encryption key used for all RADIUS communications between the network access server and the RADIUS daemon.

The following example shows how to send a request to the RADIUS server to grant a user named “pat” reverse Telnet access at port tty2 on the network access server named “maple”:

```
Username = “pat”
Password = “goaway”
User-Service-Type = Shell-User
cisco-avpair = “raccess:port#1=maple/tty2”
```

The syntax "raccess:port=any/any" permits a user to have unconditional access to network access server ports for reverse Telnet. If no "raccess:port={*nasname* }/{*tty number* }" clause exists in the user profile, the user is denied access to reverse Telnet on all ports.

For more information about configuring RADIUS, refer to the chapter “Configuring RADIUS.”



CHAPTER 79

Configuring Accounting

The AAA accounting feature allows the services that users are accessing and the amount of network resources that users are consuming to be tracked. When AAA accounting is enabled, the network access server reports user activity to the TACACS+ or RADIUS security server (depending on which security method is implemented) in the form of accounting records. Each accounting record contains accounting attribute-value (AV) pairs and is stored on the security server. This data can then be analyzed for network management, client billing, and auditing.

Prerequisites for Configuring Accounting

The following tasks must be performed before configuring accounting using named method lists:

- Enable AAA on the network access server by using the **aaa new-model** command in global configuration mode.
- Define the characteristics of the RADIUS or TACACS+ security server if RADIUS or TACACS+ authorization is issued. For more information about configuring the Cisco network access server to communicate with the RADIUS security server, see the Configuring RADIUS module. For more information about configuring the Cisco network access server to communicate with the TACACS+ security server, see the Configuring TACACS+ module.

Restrictions for Configuring Accounting

- Accounting information can be sent simultaneously to a maximum of only four AAA servers.

Information About Configuring Accounting

Named Method Lists for Accounting

Similar to authentication and authorization method lists, method lists for accounting define the way accounting is performed and the sequence in which these methods are performed.

Named accounting method lists allow particular security protocol to be designated and used on specific lines or interfaces for accounting services. The only exception is the default method list (which is named default). The default method list is automatically applied to all interfaces except those that have a named method list explicitly defined. A defined method list overrides the default method list.

A method list is simply a named list describing the accounting methods to be queried (such as RADIUS or TACACS+), in sequence. Method lists allow one or more security protocols to be designated and used for accounting, thus ensuring a backup system for accounting in case the initial method fails. Cisco IOS software uses the first method listed to support accounting; if that method fails to respond, the Cisco IOS software selects the next accounting method listed in the method list. This process continues until there is successful communication with a listed accounting method, or all methods defined are exhausted.

Note The Cisco IOS software attempts accounting with the next listed accounting method only when there is no response from the previous method. If accounting fails at any point in this cycle--meaning that the security server responds by denying the user access--the accounting process stops and no other accounting methods are attempted.

Accounting method lists are specific to the type of accounting being requested. AAA supports seven different types of accounting:

- **EXEC**: Provides information about user EXEC terminal sessions of the network access server.
- **Commands**: Provides information about the EXEC mode commands that a user issues. Command accounting generates accounting records for all EXEC mode commands, including global configuration commands, associated with a specific privilege level.
- **Connection**: Provides information about all outbound connections made from the network access server, such as Telnet, local-area transport (LAT), TN3270, packet assembler/disassembler (PAD), and rlogin.
- **System**: Provides information about system-level events.
- **Resource**: Provides “start” and “stop” records for calls that have passed user authentication, and provides “stop” records for calls that fail to authenticate.
- **VRRS**: Provides information about Virtual Router Redundancy Service (VRRS).

Note System accounting does not use named accounting lists; only the default list for system accounting can be defined.

When a named method list is created, a particular list of accounting methods for the indicated accounting type are defined.

Accounting method lists must be applied to specific lines or interfaces before any of the defined methods are performed. The only exception is the default method list (which is named "default"). If the **aaa accounting** command for a particular accounting type is issued without specifying a named method list, the default method list is automatically applied to all interfaces or lines except those that have a named method list explicitly defined (A defined method list overrides the default method list). If no default method list is defined, then no accounting takes place.

This section includes the following subsections:

Method Lists and Server Groups

A server group is a way to group existing RADIUS or TACACS+ server hosts for use in method lists. The figure below shows a typical AAA network configuration that includes four security servers: R1 and R2 are RADIUS servers, and T1 and T2 are TACACS+ servers. R1 and R2 comprise the group of RADIUS servers. T1 and T2 comprise the group of TACACS+ servers.

In Cisco IOS software, RADIUS and TACACS+ server configurations are global. A subset of the configured server hosts can be specified using server groups. These server groups can be used for a particular service. For example, server groups allow R1 and R2 to be defined as separate server groups (SG1 and SG2), and T1 and T2 as separate server groups (SG3 and SG4). This means either R1 and T1 (SG1 and SG3) or R2 and T2 (SG2 and SG4) can be specified in the method list, which provides more flexibility in the way that RADIUS and TACACS+ resources are assigned.

Server groups also can include multiple host entries for the same server, as long as each entry has a unique identifier. The combination of an IP address and a UDP port number creates a unique identifier, allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. In other words, this unique identifier enables RADIUS requests to be sent to different UDP ports on a server from the same IP address. If two different host entries on the same RADIUS server are configured for the same service: for example, accounting; the second host entry configured acts as failover backup to the first one. Using this example, if the first host entry fails to provide accounting services, the network access server tries the second host entry configured on the same device for accounting services (The RADIUS host entries are tried in the order in which they are configured).

For more information about configuring server groups and about configuring server groups based on Dialed Number Identification Service (DNIS) numbers, see the "Configuring RADIUS" or "Configuring TACACS+" modules.

AAA Accounting Methods

The following two methods of accounting are supported:

- TACACS+: The network access server reports user activity to the TACACS+ security server in the form of accounting records. Each accounting record contains accounting AV pairs and is stored on the security server.
- RADIUS: The network access server reports user activity to the RADIUS security server in the form of accounting records. Each accounting record contains accounting AV pairs and is stored on the security server.

Note Passwords and accounting logs are masked before being sent to the TACACS+ or RADIUS security servers. Use the **aaa accounting commands visible-keys** command to send unmasked information to the TACACS+ or RADIUS security servers.

Accounting Record Types

For minimal accounting, use the **stop-only** keyword, which instructs the specified method (**RADIUS** or **TACACS+**) to send a stop record accounting notice at the end of the requested user process. For more accounting information, use the **start-stop** keyword to send a start accounting notice at the beginning of the requested event and a stop accounting notice at the end of the event. To stop all accounting activities on this line or interface, use the **none** keyword.

Accounting Methods

The table below lists the supported accounting methods.

Table 87: AAA Accounting Methods

Keyword	Description
group radius	Uses the list of all RADIUS servers for accounting.
group tacacs+	Uses the list of all TACACS+ servers for accounting.
group *group-name*	Uses a subset of RADIUS or TACACS+ servers for accounting as defined by the server group *group-name*.

The method argument refers to the actual method the authentication algorithm tries. Additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all other methods return an error, specify additional methods in the command. For example, to create a method list named acct_tac1 that specifies RADIUS as the backup method of authentication in the event that TACACS+ authentication returns an error, enter the following command:

```
aaa accounting network acct_tac1 stop-only group tacacs+ group radius
```

To create a default list that is used when a named list is not specified in the **aaa accounting**command, use the **default** keyword followed by the methods that are wanted to be used in default situations. The default method list is automatically applied to all interfaces.

For example, to specify RADIUS as the default method for user authentication during login, enter the following command:

```
aaa accounting network default stop-only group radius
```

AAA Accounting supports the following methods:

- **group tacacs** : To have the network access server send accounting information to a TACACS+ security server, use the **group tacacs+** *method* keyword.
- **group radius** : To have the network access server send accounting information to a RADIUS security server, use the **group radius** *method* keyword.
- **group** *group-name* : To specify a subset of RADIUS or TACACS+ servers to use as the accounting method, use the **aaa accounting**command with the **group** *group-name* method. To specify and define the group name and the members of the group, use the **aaa group server** command. For example, use the **aaa group server** command to first define the members of **group loginrad**:

```
aaa group server radius loginrad
 server 172.16.2.3
 server 172.16.2 17
 server 172.16.2.32
```

This command specifies RADIUS servers 172.16.2.3, 172.16.2.17, and 172.16.2.32 as members of the **group loginrad**.

To specify **group loginrad** as the method of network accounting when no other method list has been defined, enter the following command:

```
aaa accounting network default start-stop group loginrad
```

Before a group name can be used as the accounting method, communication with the RADIUS or TACACS+ security server must be enabled.

AAA Accounting Types

EXEC Accounting

EXEC accounting provides information about user EXEC terminal sessions (user shells) on the network access server, including username, date, start and stop times, the access server IP address, and (for dial-in users) the telephone number the call originated from.

The following example shows the information contained in a RADIUS EXEC accounting record for a dial-in user:

```
Wed Jun 27 04:26:23 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 1
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329483"
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Exec-User
        Acct-Session-Id = "00000006"
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
Wed Jun 27 04:27:25 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 1
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329483"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Exec-User
        Acct-Session-Id = "00000006"
        Acct-Session-Time = 62
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
```

The following example shows the information contained in a TACACS+ EXEC accounting record for a dial-in user:

```
Wed Jun 27 03:46:21 2001        172.16.25.15    username1   tty3    5622329430/4327528
start
 task_id=2       service=shell
Wed Jun 27 04:08:55 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop
 task_id=2       service=shell   elapsed_time=1354
```

The following example shows the information contained in a RADIUS EXEC accounting record for a Telnet user:

```
Wed Jun 27 04:48:32 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 26
        User-Name = "username1"
        Caller-ID = "10.68.202.158"
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Exec-User
        Acct-Session-Id = "00000010"
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"

Wed Jun 27 04:48:46 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 26
        User-Name = "username1"
        Caller-ID = "10.68.202.158"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Exec-User
        Acct-Session-Id = "00000010"
        Acct-Session-Time = 14
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
```

The following example shows the information contained in a TACACS+ EXEC accounting record for a Telnet user:

```
Wed Jun 27 04:06:53 2001        172.16.25.15    username1   tty26   10.68.202.158
starttask_id=41      service=shell
Wed Jun 27 04:07:02 2001        172.16.25.15    username1   tty26   10.68.202.158
stoptask_id=41       service=shell   elapsed_time=9
```

Command Accounting

Command accounting provides information about the EXEC shell commands for a specified privilege level that are being executed on a network access server. Each command accounting record includes a list of the commands executed for that privilege level, as well as the date and time each command was executed, and the user who executed it.

The following example shows the information contained in a TACACS+ command accounting record for privilege level 1:

```
Wed Jun 27 03:46:47 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=3       service=shell   priv-lvl=1      cmd=show version <cr>
Wed Jun 27 03:46:58 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=4       service=shell   priv-lvl=1      cmd=show interfaces Ethernet 0
<cr>
Wed Jun 27 03:47:03 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=5       service=shell   priv-lvl=1      cmd=show ip route <cr>
```

The following example shows the information contained in a TACACS+ command accounting record for privilege level 15:

```
Wed Jun 27 03:47:17 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=6       service=shell   priv-lvl=15     cmd=configure terminal <cr>
Wed Jun 27 03:47:21 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=7       service=shell   priv-lvl=15     cmd=interface Serial 0 <cr>
```

```
Wed Jun 27 03:47:29 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop    task_id=8      service=shell   priv-lvl=15    cmd=ip address 10.1.1.1 255.255.255.0
 <cr>
```

Note The Cisco implementation of RADIUS does not support command accounting.

Connection Accounting

Connection accounting provides information about all outbound connections made from the network access server such as Telnet, LAT, TN3270, PAD, and rlogin.

The following example shows the information contained in a RADIUS connection accounting record for an outbound Telnet connection:

```
Wed Jun 27 04:28:00 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 2
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329477"
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Login
        Acct-Session-Id = "00000008"
        Login-Service = Telnet
        Login-IP-Host = "10.68.202.158"
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"

Wed Jun 27 04:28:39 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 2
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329477"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Login
        Acct-Session-Id = "00000008"
        Login-Service = Telnet
        Login-IP-Host = "10.68.202.158"
        Acct-Input-Octets = 10774
        Acct-Output-Octets = 112
        Acct-Input-Packets = 91
        Acct-Output-Packets = 99
        Acct-Session-Time = 39
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
```

The following example shows the information contained in a TACACS+ connection accounting record for an outbound Telnet connection:

```
Wed Jun 27 03:47:43 2001        172.16.25.15    username1   tty3    5622329430/4327528
start    task_id=10     service=connection     protocol=telnet addr=10.68.202.158 cmd=telnet
 username1-sun
Wed Jun 27 03:48:38 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop    task_id=10     service=connection     protocol=telnet addr=10.68.202.158 cmd=telnet
```

```
 username1-sun     bytes_in=4467   bytes_out=96    paks_in=61      paks_out=72 elapsed_time=55
```

The following example shows the information contained in a RADIUS connection accounting record for an outbound rlogin connection:

```
Wed Jun 27 04:29:48 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 2
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329477"
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Login
        Acct-Session-Id = "0000000A"
        Login-Service = Rlogin
        Login-IP-Host = "10.68.202.158"
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"

Wed Jun 27 04:30:09 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 2
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329477"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Login
        Acct-Session-Id = "0000000A"
        Login-Service = Rlogin
        Login-IP-Host = "10.68.202.158"
        Acct-Input-Octets = 18686
        Acct-Output-Octets = 86
        Acct-Input-Packets = 90
        Acct-Output-Packets = 68
        Acct-Session-Time = 22
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
```

The following example shows the information contained in a TACACS+ connection accounting record for an outbound rlogin connection:

```
Wed Jun 27 03:48:46 2001        172.16.25.15    username1   tty3    5622329430/4327528
start    task_id=12     service=connection     protocol=rlogin addr=10.68.202.158 cmd=rlogin
 username1-sun /user username1
Wed Jun 27 03:51:37 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=12     service=connection     protocol=rlogin addr=10.68.202.158 cmd=rlogin
 username1-sun /user username1 bytes_in=659926 bytes_out=138   paks_in=2378    paks_
out=1251        elapsed_time=171
```

The following example shows the information contained in a TACACS+ connection accounting record for an outbound LAT connection:

```
Wed Jun 27 03:53:06 2001        172.16.25.15    username1   tty3    5622329430/4327528
start    task_id=18      service=connection      protocol=lat    addr=VAX        cmd=lat
VAX
Wed Jun 27 03:54:15 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=18      service=connection      protocol=lat    addr=VAX        cmd=lat
VAX  bytes_in=0      bytes_out=0     paks_in=0      paks_out=0      elapsed_time=6
```

System Accounting

System accounting provides information about all system-level events (for example, when the system reboots or when accounting is turned on or off).

The following accounting record shows a typical TACACS+ system accounting record server indicating that AAA Accounting has been turned off:

```
Wed Jun 27 03:55:32 2001        172.16.25.15    unknown unknown unknown start   task_id=25
   service=system
event=sys_acct  reason=reconfigure
```

Note The precise format of accounting packets records may vary depending on the TACACS+ daemon.

The following accounting record shows a TACACS+ system accounting record indicating that AAA Accounting has been turned on:

```
Wed Jun 27 03:55:22 2001        172.16.25.15    unknown unknown unknown stop    task_id=23
   service=system
event=sys_acct  reason=reconfigure
```

AAA Accounting Enhancements

AAA Broadcast Accounting

AAA broadcast accounting allows accounting information to be sent to multiple AAA servers at the same time; that is, accounting information can be broadcast to one or more AAA servers simultaneously. This functionality allows service providers to send accounting information to their own private AAA servers and to the AAA servers of their end customers. It also provides redundant billing information for voice applications.

Broadcasting is allowed among groups of RADIUS or TACACS+ servers, and each server group can define its backup servers for failover independently of other groups.

Thus, service providers and their end customers can use different protocols (RADIUS or TACACS+) for the accounting server. Service providers and their end customers can also specify their backup servers independently. As for voice applications, redundant accounting information can be managed independently through a separate group with its own failover sequence.

AAA Session MIB

The AAA session MIB feature allows customers to monitor and terminate their authenticated client connections using Simple Network Management Protocol (SNMP). The data of the client is presented so that it correlates directly to the AAA Accounting information reported by either the RADIUS or the TACACS+ server. AAA session MIB provides the following information:

- Statistics for each AAA function (when used in conjunction with the **show radius statistics** command)
- Status of servers providing AAA functions
- Identities of external AAA servers
- Real-time information (such as idle times), providing additional criteria for use by SNMP networks for assessing whether or not to terminate an active call

The table below shows the SNMP user-end data objects that can be used to monitor and terminate authenticated client connections with the AAA session MIB feature.

Table 88: SNMP End-User Data Objects

SessionId	The session identification used by the AAA Accounting protocol (same value as reported by RADIUS attribute 44 (Acct-Session-ID)).
UserId	The user login ID or zero-length string if a login is unavailable.
IpAddr	The IP address of the session or 0.0.0.0 if an IP address is not applicable or unavailable.
IdleTime	The elapsed time in seconds that the session has been idle.
Disconnect	The session termination object used to disconnect the given client.
CallId	The entry index corresponding to this accounting session that the Call Tracker record stored.

The table below describes the AAA summary information provided by the AAA session MIB feature using SNMP on a per-system basis.

Table 89: SNMP AAA Session Summary

ActiveTableEntries	Number of sessions currently active.
ActiveTableHighWaterMark	Maximum number of sessions present at once since last system reinstallation.
TotalSessions	Total number of sessions since last system reinstallation.
DisconnectedSessions	Total number of sessions that have been disconnected using since last system reinstallation.

Accounting Attribute-Value Pairs

The network access server monitors the accounting functions defined in either TACACS+ AV pairs or RADIUS attributes, depending on which security method is implemented.

How to Configure AAA Accounting

Configuring AAA Accounting Using Named Method Lists

To configure AAA Accounting using named method lists, perform the following steps:

Note System accounting does not use named method lists. For system accounting, define only the default method list.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa accounting** {**system** | **network** | **exec** | **connection** | **commands** *level*} {**default** | *list-name*} {**start-stop** | **stop-only** | **none**} [*method1* [*method2...*]] **Example:** `Device(config)# aaa accounting system default start-stop`	Creates an accounting method list and enables accounting. The argument *list-name* is a character string used to name the created list.
Step 4	Do one of the following: • **line** [**aux** | **console** | **tty** | **vty**] *line-number* [*ending-line-number*] • **interface** *interface-type interface-number* **Example:** `Device(config)# line aux line1`	Enters the line configuration mode for the lines to which the accounting method list is applied. or Enters the interface configuration mode for the interfaces to which the accounting method list is applied.
Step 5	**accounting** {**arap** | **commands** *level* | **connection** | **exec**} {**default** | *list-name*} **Example:** `Device(config-line)# accounting arap default`	Applies the accounting method list to a line or set of lines.
Step 6	**end** **Example:** `Device(config-line)# end`	(Optional) Exits line configuration mode and returns to privileged EXEC mode.

Suppressing Generation of Accounting Records for Null Username Sessions

When AAA Accounting is activated, the Cisco IOS software issues accounting records for all users on the system, including users whose username string, because of protocol translation, is NULL. An example of this is users who come in on lines where the **aaa authentication login** *method-list* **none** command is applied. To prevent accounting records from being generated for sessions that do not have usernames associated with them, use the following command in global configuration mode:

Command	Purpose
`Device(config)# aaa accounting suppress null-username`	Prevents accounting records from being generated for users whose username string is NULL.

Generating Interim Accounting Records

To enable periodic interim accounting records to be sent to the accounting server, use the following command in global configuration mode:

Command	Purpose
`Device(config)# aaa accounting update [newinfo] [periodic] number`	Enables periodic interim accounting records to be sent to the accounting server.

When the **aaa accounting update** command is activated, the Cisco IOS software issues interim accounting records for all users on the system. If the keyword **newinfo** is used, interim accounting records are sent to the accounting server every time there is new accounting information to report. An example of this would be when IPCP completes IP address negotiation with the remote peer. The interim accounting record includes the negotiated IP address used by the remote peer.

When used with the keyword **periodic**, interim accounting records are sent periodically as defined by the *number* argument. The interim accounting record contains all of the accounting information recorded for that user up to the time the interim accounting record is sent.

Caution Using the **aaa accounting update periodic** command can cause heavy congestion when many users are logged in to the network.

Configuring an Alternate Method to Enable Periodic Accounting Records

You can use the following alternative method to enable periodic interim accounting records to be sent to the accounting server.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa accounting network default** **Example:** `Device(config)# aaa accounting network default`	Configures the default accounting for all network-related service requests and enters accounting method list configuration mode.
Step 4	**action-type** {**none** | **start-stop** [**periodic** {**disable** | **interval** *minutes*}] | **stop-only**} **Example:** `Device(cfg-acct-mlist)# action-type start-stop periodic interval 5`	Specifies the type of action to be performed on accounting records. • (Optional) The periodic keyword specifies periodic accounting action. • The interval keyword specifies the periodic accounting interval. • The *value* argument specifies the intervals for accounting update records (in minutes). • The **disable** keyword disables periodic accounting.
Step 5	**end** **Example:** `Device(cfg-acct-mlist)# end`	Exits accounting method list configuration mode and returns to privileged EXEC mode.

Generating Interim Service Accounting Records

Perform this task to enable the generation of interim service accounting records at periodic intervals for subscribers.

Before you begin

RADIUS Attribute 85 in the user service profile always takes precedence over the configured interim-interval value. RADIUS Attribute 85 must be in the user service profile. See the RADIUS Attributes Overview and RADIUS IETF Attributes feature document for more information.

Note If RADIUS Attribute 85 is not in the user service profile, then the interim-interval value configured in Generating Interim Accounting Records is used for service interim accounting records.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**subscriber service accounting interim-interval** *minutes* **Example:** `Device(config)# subscriber service accounting interim-interval 10`	Enables the generation of interim service accounting records at periodic intervals for subscribers. The *minutes* argument indicates the number of periodic intervals to send accounting update records from 1 to 71582 minutes.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Generating Accounting Records for a Failed Login or Session

When AAA accounting is activated, the Cisco IOS XE software does not generate accounting records for system users who fail login authentication, or who succeed in login authentication.

To specify that accounting stop records be generated for users who fail to authenticate at login or during session negotiation, use the following command in global configuration mode:

Command or Action	Purpose
aaa accounting send stop-record authentication failure	Generates “stop” records for users who fail to authenticate at login or during session negotiation.

Suppressing System Accounting Records over Switchover

To suppress the system accounting-on and accounting-off messages during switchover, use the following command in global configuration mode:

Command or Action	Purpose
aaa accounting redundancy suppress system-records	Suppresses the system accounting messages during switchover.

Configuring AAA Resource Failure Stop Accounting

To enable resource failure stop accounting, use the following command in global configuration mode:

Command	Purpose
`Device(config)# aaa accounting resource` *method-list* **stop-failure group** *server-group*	Generates a "stop" record for any calls that do not reach user authentication. **Note** Before configuring this feature, the tasks described in the Prerequisites for Configuring Accounting, on page 1099 section must be performed, and SNMP must be enabled on the network access server.

Configuring AAA Resource Accounting for Start-Stop Records

To enable full resource accounting for start-stop records, use the following command in global configuration mode:

Command	Purpose
`Device(config)# aaa accounting resource` *method-list* **start-stop group** *server-group*	Supports the ability to send a "start" record at each call setup. followed with a corresponding "stop" record at the call disconnect. **Note** Before configuring this feature, the tasks described in the Prerequisites for Configuring Accounting, on page 1099 section must be performed, and SNMP must be enabled on the network access server.

AAA Broadcast Accounting

AAA broadcast accounting allows accounting information to be sent to multiple AAA servers at the same time; that is, accounting information can be broadcast to one or more AAA servers simultaneously. This functionality allows service providers to send accounting information to their own private AAA servers and to the AAA servers of their end customers. It also provides redundant billing information for voice applications.

Broadcasting is allowed among groups of RADIUS or TACACS+ servers, and each server group can define its backup servers for failover independently of other groups.

Thus, service providers and their end customers can use different protocols (RADIUS or TACACS+) for the accounting server. Service providers and their end customers can also specify their backup servers independently. As for voice applications, redundant accounting information can be managed independently through a separate group with its own failover sequence.

Configuring Per-DNIS AAA Broadcast Accounting

To configure AAA broadcast accounting per DNIS, use the **aaa dnis map accounting network**command in global configuration mode:

Command	Purpose
`Device(config)# aaa dnis map dnis-number accounting network [start-stop | stop-only | none] [broadcast] method1 [method2...]`	Allows per-DNIS accounting configuration. This command has precedence over the global **aaa accounting** command. Enables sending accounting records to multiple AAA servers. Simultaneously sends accounting records to the first server in each group. If the first server is unavailable, failover occurs using the backup servers defined within that group.

Establishing a Session with a Device if the AAA Server Is Unreachable

To establish a console session with a device if the AAA server is unreachable, use the following command in global configuration mode:

Command or Action	Purpose
no aaa accounting system guarantee-first	The **aaa accounting system guarantee-first** command guarantees system accounting as the first record, which is the default condition. In some situations, users may be prevented from starting a session on the console or terminal connection until after the system reloads, which can take more than three minutes. To resolve this problem, use the **no aaa accounting system guarantee-first** command.

Monitoring Accounting

No specific **show** command exists for either RADIUS or TACACS+ accounting. To obtain accounting records displaying information about users logged in, use the following command in privileged EXEC mode:

Command or Action	Purpose
show accounting	Allows display of the active accountable events on the network and helps collect information in the event of a data loss on the accounting server.

Troubleshooting Accounting

To troubleshoot accounting information, use the following command in privileged EXEC mode:

Command or Action	Purpose
debug aaa accounting	Displays information on accountable events as they occur.

Configuration Examples for AAA Accounting

Example: Configuring a Named Method List

The following example shows how to configure a Cisco device (enabled for AAA and communication with a RADIUS security server) in order for AAA services to be provided by the RADIUS server. If the RADIUS server fails to respond, then the local database is queried for authentication and authorization information, and accounting services are handled by a TACACS+ server.

```
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication login admins local
Device(config)# aaa authorization network network1 group radius local
Device(config)# aaa accounting network network2 start-stop group radius group tacacs+
Device(config)# username root password ALongPassword
Device(config)# tacacs server server1
Device(config-server-tacacs)# address ipv4 172.31.255.0
Device(config-server-tacacs)# key goaway
Device(config-server-tacacs)# exit
Device(config)# radius server isp
Device(config-sg-radius)# key myRaDiUSpassWoRd
Device(config-sg-radius)# exit
Device(config)# line 1 16
Device(config-line)# autoselect during-login
Device(config-line)# login authentication admins
Device(config-line)# modem dialin
Device(config-line)# end
```

The lines in this sample RADIUS AAA configuration are defined as follows:

- The **aaa new-model** command enables AAA network security services.
- The **aaa authentication login admins local** command defines a method list, “admins”, for login authentication.
- The **tacacs server** command defines the name of the TACACS+ server host.
- The **key** command defines the shared secret text string between the network access server and the TACACS+ server host.
- The **radius server** command defines the name of the RADIUS server host.
- The **key** command defines the shared secret text string between the network access server and the RADIUS server host.
- The **interface group-async** command selects and defines an asynchronous interface group.
- The **group-range** command defines the member asynchronous interfaces in the interface group.
- The **line** command switches the configuration mode from global configuration to line configuration and identifies the specific lines being configured.
- The **login authentication admins** command applies the admins method list for login authentication.
- The **modem dialin** command configures modems attached to the selected lines to accept only incoming calls.

The table below describes the fields contained in the preceding output.

Table 90: show accounting Field Descriptions

Field	Description
Active Accounted actions on	Terminal line or interface name user with which the user logged in.
User	User's ID.
Priv	User's privilege level.
Task ID	Unique identifier for each accounting session.
Accounting Record	Type of accounting session.
Elapsed	Length of time (hh:mm:ss) for this session type.
attribute=value	AV pairs associated with this accounting session.

Example: Configuring AAA Broadcast Accounting

The following example shows how to turn on broadcast accounting using the global **aaa accounting** command:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius isp
Device(config-sg-radius)# server 10.0.0.1
Device(config-sg-radius)# server 10.0.0.2
Device(config-sg-radius)# exit
Device(config)# aaa group server tacacs+ isp_customer
Device config-sg-tacacs+)# server 172.0.0.1
Device config-sg-tacacs+)# exit
Device(config)# aaa accounting network default start-stop broadcast group isp group
isp_customer
Device(config)# tacacs server server1
Device(config-server-tacacs)# address ipv4 172.0.0.1
Device(config-server-tacacs)# key key2
Device(config-server-tacacs)# end
```

The **broadcast** keyword causes "start" and "stop" accounting records for network connections to be sent simultaneously to server 10.0.0.1 in the group isp and to server 172.0.0.1 in the group isp_customer. If server 10.0.0.1 is unavailable, failover to server 10.0.0.2 occurs. If server 172.0.0.1 is unavailable, no failover occurs because backup servers are not configured for the group isp_customer.

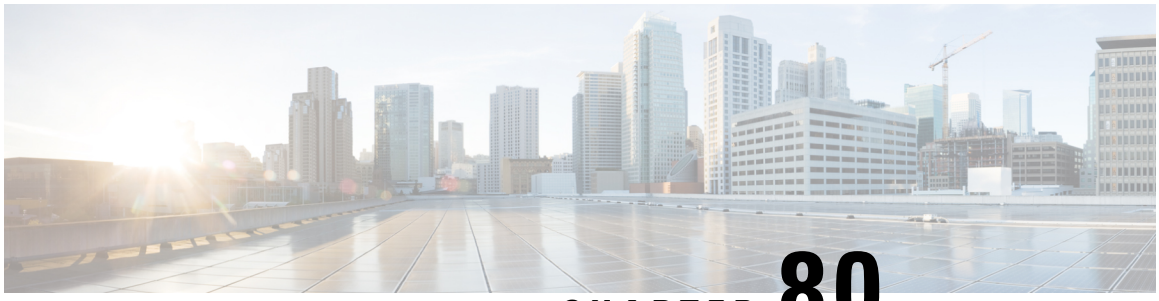
Example: Configuring per-DNIS AAA Broadcast Accounting

The following example shows how to turn on per-DNIS broadcast accounting using the global **aaa dnis map accounting network** command:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius isp
Device(config-sg-radius)# server 10.0.0.1
Device(config-sg-radius)# server 10.0.0.2
Device(config-sg-radius)# exit
Device(config)# aaa group server tacacs+ isp_customer
```

```
Device config-sg-tacacs+)# server 172.0.0.1
Device config-sg-tacacs+)# exit
Device(config)# aaa dnis map enable
Device(config)# aaa dnis map 7777 accounting network start-stop broadcast group isp group
isp_customer
Device(config)# tacacs server server1
Device(config-server-tacacs)# address ipv4 172.0.0.1
Device(config-server-tacacs)# key key_2
Device(config-server-tacacs)# end
```

The **broadcast** keyword causes “start” and “stop” accounting records for network connection calls having DNIS number 7777 to be sent simultaneously to server 10.0.0.1 in the group isp and to server 172.0.0.1 in the group isp_customer. If server 10.0.0.1 is unavailable, failover to server 10.0.0.2 occurs. If server 172.0.0.1 is unavailable, no failover occurs because backup servers are not configured for the group isp_customer.



CHAPTER 80

Configuring Local Authentication and Authorization

How to Configure Local Authentication and Authorization

This section provides information about the task that comprise local authentication and authorization configuration.

Configuring the Switch for Local Authentication and Authorization

You can configure AAA to operate without a server by setting the switch to implement AAA in local mode. The switch then handles authentication and authorization. No accounting is available in this configuration.

Note To secure the switch for HTTP access by using AAA methods, you must configure the switch with the **ip http authentication aaa** global configuration command. Configuring AAA authentication does not secure the switch for HTTP access by using AAA methods.

Follow these steps to configure AAA to operate without a server by setting the switch to implement AAA in local mode:

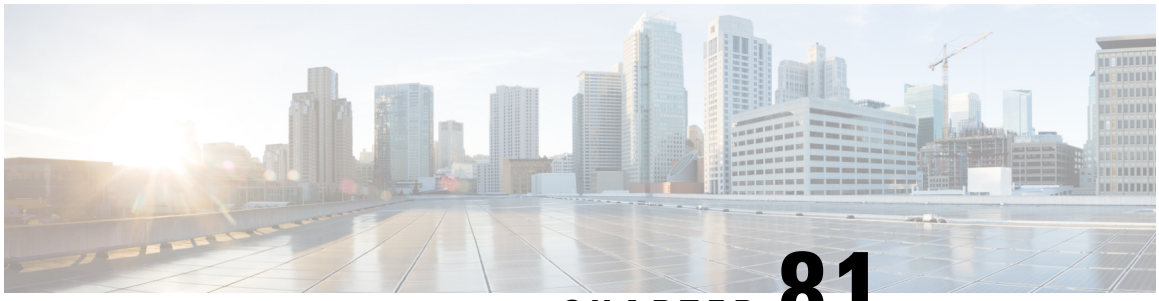
Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**aaa authentication login default local** **Example:** `Device(config)# aaa authentication login default local`	Sets the login authentication to use the local username database. The **default** keyword applies the local user database authentication to all ports.
Step 5	**aaa authorization exec default local** **Example:** `Device(config)# aaa authorization exec default local`	Configures user AAA authorization, check the local database, and allow the user to run an EXEC shell.
Step 6	**aaa authorization network default local** **Example:** `Device(config)# aaa authorization network default local`	Configures user AAA authorization for all network-related service requests.
Step 7	**username** *name* [**privilege** *level*] {**password** *encryption-type password*} **Example:** `Device(config)# username your_user_name privilege 1 password 7 secret567`	Enters the local database, and establishes a username-based authentication system. Repeat this command for each user. • For *name*, specify the user ID as one word. Spaces and quotation marks are not allowed. • (Optional) For *level*, specify the privilege level the user has after gaining access. The range is 0 to 15. Level 15 gives privileged EXEC mode access. Level 0 gives user EXEC mode access. • For *encryption-type*, enter 0 to specify that an unencrypted password follows. Enter 7 to specify that a hidden password follows. • For *password*, specify the password the user must enter to gain access to the switch. The password must be from 1 to 25 characters, can contain embedded spaces, and must be the last option specified in the **username** command.
Step 8	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Monitoring Local Authentication and Authorization

To display Local Authentication and Authorization configuration, use the **show running-config** command in privileged EXEC mode.



CHAPTER 81

Configuring AAA Authorization and Authentication Cache

The AAA Authorization and Authentication Cache feature allows you to cache authorization and authentication responses for a configured set of users or service profiles, providing performance improvements and an additional level of network reliability. Users and service profiles that are returned from authorization and authentication responses can be queried from multiple sources and need not depend solely on an offload server. This feature also provides a failover mechanism so that if a network RADIUS or TACACS+ server is unable to provide authorization and authentication responses network users and administrators can still access the network.

Prerequisites for Implementing Authorization and Authentication Profile Caching

The following prerequisites apply to implementing authorization and authentication profile caching:

- Understand how you want to implement profile caching, that is, are profiles being cached to improve network performance or as a failover mechanism if your network authentication and authorization servers (RADIUS and TACACS+) become unavailable.
- RADIUS and TACACS+ server groups must already be configured.

Information About Implementing Authorization and Authentication Profile Caching

The following sections provide information about implementing authorization and authentication profile caching.

Network Performance Optimization Using Authorization and Authentication Profile Caching

RADIUS and TACACS+ clients run on Cisco devices and send authentication requests to a central RADIUS or TACACS+ server that contains all user authentication and network service access information. The device is required to communicate with an offload RADIUS or TACACS+ server to authenticate a given call and then apply a policy or service to that call. Unlike authentication, authorization, and accounting (AAA) accounting, AAA authentication and authorization is a blocking procedure, which means the call setup may not proceed while the call is being authenticated and authorized. Thus, the time required to process the call setup is directly impacted by the time required to process such an authentication or authorization request from the device to the offload RADIUS or TACACS+ server, and back again. Any communication problems in the transmission, offload server utilization, and numerous other factors cause significant degradation in a device's call setup performance because of the AAA authentication and authorization step. The problem is further highlighted when multiple AAA authentications and authorizations are needed for a single call or session.

A solution to this problem is to minimize the impact of such authentication requests by caching the authentication and authorization responses for specific users on the device, thereby removing the need to send the requests to an offload server again and again. This profile caching adds significant performance improvements to the call setup times. Profile caching also provides an additional level of network reliability because user and service profiles that are returned from authentication and authorization responses can be queried from multiple sources and need not depend solely on an offload server.

To take advantage of this performance optimization, you need to configure the authentication method list so that the AAA cache profile is queried first when a user attempts to authenticate to the device. See Method Lists in Authorization and Authentication Profile Caching section for more information.

Authorization and Authentication Profile Caching as a Failover Mechanism

If, for whatever reason, RADIUS or TACACS+ servers are unable to provide authentication and authorization responses, network users and administrators can be locked out of the network. The profile caching feature allows usernames to be authorized without having to complete the authentication phase. For example, a user by the name *user100@example.com* with the password *secretpassword1* can be stored in a profile cache using the regular expression .*@example.com. Another user by the name *user101@example.com* with the password *secretpassword2* can also be stored using the same regular expression, and so on. Because the number of users in the .*@example.com profile could run into thousands, it is not feasible to authenticate each user with their personal password. Therefore, authentication is disabled, and each user simply accesses authorization profiles from a common Access Response stored in the cache.

The same reasoning applies to cases involving higher-end security mechanisms, such as Extensible Authentication Protocol (EAP), which utilize an encrypted password for communication between the client and the AAA offload server. To allow these unique secure username and password profiles to retrieve their authorization profiles, authentication is bypassed.

To take advantage of this failover capability, you need to configure the authentication and authorization method list so that the cache server group is queried last when a user attempts to authenticate to the device. See Method Lists in Authorization and Authentication Profile Caching section for more information.

Method Lists in Authorization and Authentication Profile Caching

A method list is a sequential list describing the authentication methods to be queried in order to authenticate a user. Cisco support methods such as local (use the local Cisco IOS XE database), none (do nothing), RADIUS server group, or TACACS+ server group. Typically, more than one method can be configured into a method list. Cisco IOS XE software uses the first listed method to authenticate users. If that method fails to respond, the Cisco IOS XE software selects the next authentication method listed in the method list. This process continues until there is successful communication with a listed authentication method, or until all the methods defined in the method list are exhausted.

To optimize network performance or provide failover capability using the profile caching feature, change the order of the authentication and authorization methods in the method list. To optimize network performance, make sure the cache server group appears first in the method list. For failover capability, the cache server group should appear last in the method list.

Authorization and Authentication Profile Caching Guidelines

Because the number of usernames and profiles that can request to be authenticated or authorized at a given device on a given point of presence (POP) can be quite extensive, it is not feasible to cache all of them. Therefore, only usernames and profiles that are commonly used or that share a common authentication and authorization response should be configured to use caching. Commonly used usernames such as aolip and aolnet, which are used for America Online (AOL) calls, or preauthentication dialed number identification service (DNIS) numbers used to connect Public Switched Telephone Network (PSTN) calls to a network-attached storage device, along with domain-based service profiles, are all examples of usernames and profiles that can benefit from authentication and authorization caching.

General Configuration Procedure for Implementing Authorization and Authentication Profile Caching

To implement authorization and authentication profile caching, complete the following procedure:

1. Create cache profile groups and define the rules for what information is cached in each group.

   Entries that match based on exact username and regular expressions, or specify all authentication and authorization requests, can be cached.

2. Update existing server groups to reference newly defined cache groups.
3. Update authentication or authorization method lists to use the cached information to optimize network performance or provide a failover mechanism.

How to Implement Authorization and Authentication Profile Caching

The following sections provide information about the various tasks that comprise authorization and authentication profile-caching configuration.

Creating Cache Profile Groups and Defining Caching Rules

Perform this task to create a cache profile group, define the rules for what information is cached in that group, and verify and manage cache profile entries.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables the AAA access control model.
Step 4	**aaa cache profile** *group-name* **Example:** `Device(config)# aaa cache profile networkusers@companyname`	Defines an authentication and authorization cache profile server group and enters profile map configuration mode.
Step 5	**profile** *name* [**no-auth**] **Example:** `Device(config-profile-map)# profile networkuser1 no-auth`	Creates an individual authentication and authorization cache profile based on a username match. • The *name* argument must be an exact match to a username being queried by an authentication or authorization service request. • Use the **no-auth** keyword to bypass authentication for this user. **Note** • For EAP-PEAP with MSCHAPv2 and EAP-PEAP with GTC methods, AAA authentication caching for 802.1x is supported with or without bypass authentication. However, for EAP methods such as EAP-TLS and EAP-MD5, AAA authentication caching for 802.1x is only supported with bypass authentication.

	Command or Action	Purpose
		• For EAP-MSCHAPV2 use cases that do not use **no-auth** (bypass authentication), the administrator must configure the Cisco AV-pairs AS-username and AS-passwordHash on the Cisco Identity Services Engine (ISE), such that Cisco ISE sends these RADIUS attributes through the RADIUS `ACCESS-Accept` message to the network access server (NAS) device. Also, AS-passwordHash must be configured with nt-hash of the user password. Repeat this step for each username that you want to add to the profile group in Step 4.
Step 6	**regexp** *matchexpression* {**any** | **only**} [**no-auth**] **Example:** `Device(config-profile-map)# regexp .*@example.com any no-auth`	(Optional) Creates an entry in a cache profile group that matches a regular expression. • If you use the **any** keyword, all the unique usernames matching the regular expression are saved. • If you use the **only** keyword, only one profile entry is cached for all the usernames matching the regular expression. • Use the **no-auth** keyword to bypass authentication for a user or set of users. • Because the number of entries in a regular expression cache profile group could run into thousands, and validating each request against a regular expression can be time consuming, we recommend that you do not use regular expression entries in cache profile groups. Repeat this step for each regular expression that you want to add to the cache profile group defined in Step 4.
Step 7	**all** [**no-auth**] **Example:** `Device(config-profile-map)# all no-auth`	(Optional) Specifies that all authentication and authorization requests are cached. • Use the **all** keyword for specific service authorization requests, but avoid it when dealing with authentication requests.

	Command or Action	Purpose
Step 8	**end** **Example:** `Device(config-profile-map)# end`	Exits profile map configuration mode and returns to privileged EXEC mode.
Step 9	**show aaa cache group** *name* {**profile** *name* | **all**} **Example:** `Device# show aaa cache group networkusers@companyname all`	(Optional) Displays an individual server group profile details or all the server group profile details.
Step 10	**clear aaa cache group** *name* {**profile** *name* | **all**} **Example:** `Device# clear aaa cache group networkusers@companyname profile networkuser1`	(Optional) Clears an individual entry or all the entries in the cache.
Step 11	**debug aaa cache group** **Example:** `Device# debug aaa cache group`	(Optional) Displays debug information about cached entries.

Defining RADIUS and TACACS Server Groups that Use Cache Profile Group Information

Perform this task to define how RADIUS and TACACS+ server groups use the information stored in each cache profile group.

Before you begin

RADIUS and TACACS+ server groups must be created.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:**	Enables the AAA access control model.

	Command or Action	Purpose
	`Device(config)# aaa new-model`	
Step 4	**aaa group server radius** *group-name* or **aaa group server tacacs+** *group-name* **Example:** `Device(config)# aaa group server radius networkusers@companyname`	Enters RADIUS server group configuration mode. To enter TACACS+ server group configuration mode, use the **aaa group server tacacs+** *group-name* command.
Step 5	**cache authorization profile** *name* **Example:** `Device(config-sg-radius)# cache authorization profile networkusers@companyname`	Activates the authorization caching rules in the networkusers profile for this RADIUS or TACACS+ server group. The *name* argument is a AAA cache profile group name.
Step 6	**cache authentication profile** *name* **Example:** `Device(config-sg-radius)# cache authentication profile networkusers@companyname`	Activates the authentication-caching rules in the networkusers profile for this RADIUS or TACACS+ server group.
Step 7	**cache expiry** *hours* {**enforce** | **failover**} **Example:** `Device(config-sg-radius)# cache expiry 240 failover`	(Optional) Sets the amount of time before a cache profile entry expires (becomes stale). Use the **enforce** keyword to specify that after a cache profile entry expires, it is not used again. Use the **failover** keyword to specify that an expired cache profile entry can be used if all other methods to authenticate and authorize the user fails.
Step 8	**end** **Example:** `Device(config-sg-radius)# end`	Returns to privileged EXEC mode.

Updating Authorization and Authentication Method Lists to Specify How Cache Information is Used

Perform this task to update authorization and authentication method lists to use the authorization and authentication cache information.

Before you begin

Method lists must already be defined.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables the AAA access control model.
Step 4	**aaa authorization** {**network** | **exec** | **commands** *level* | **reverse-access** | **configuration**} {**default** | *list-name*} [*method1* [*method2...*]] **Example:** `Device(config)# aaa authorization network default cache networkusers@companyname group networkusers@companyname`	Enables AAA authorization and creates method lists, which define the authorization methods used when a user accesses a specified function.
Step 5	**aaa authentication login** {**default** | *list-name*} *method1* [*method2...*] **Example:** `Device(config)# aaa authentication login default cache adminusers group adminusers`	Sets the authentication at login.
Step 6	**aaa authentication dot1x** {**default** | *list-name*} *method1* [*method2...*] **Example:** `Device(config)# aaa authentication dot1x default cache RADGRP group RADGRP`	Sets the authentication for use on interfaces running IEEE 802.1x.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuration Examples for Implementing Authorization and Authentication Profile Caching

This following sections display configuration examples for implementing authorization and authentication profile caching.

Example: Implementing Authorization and Authentication Profile Caching for Network Optimization

The following configuration example shows how to:

- Define a cache profile group admin_users that contains the names of all the administrators on the network and sets this list as the default list that is used for all login and privileged exec sessions.
- Activate the new caching rules for a RADIUS server group.
- Add the new cache profile group in the authentication and authorization method list and change the method order so that the cache profile group is queried first.

```
configure terminal
 aaa new-model
 ! Define aaa cache profile groups and the rules for what information is saved to cache.
 aaa cache profile admin_users
 profile adminuser1
 profile adminuser2
 profile adminuser3
 profile adminuser4
 profile adminuser5
 exit
 ! Define server groups that use the cache information in each profile group.
 aaa group server radius admins@companyname.com
 cache authorization profile admin_users
 cache authentication profile admin_users
 ! Update authentication and authorization method lists to specify how profile groups and
server groups are used.
 aaa authentication login default cache admins@companyname.com group admins@companyname.com

 aaa authorization exec default cache admins@companyname.com group admins@companyname.com
 end
```

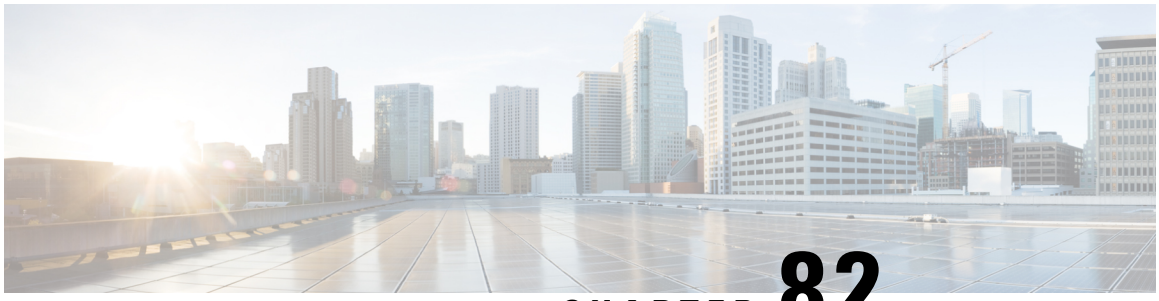
Example: Implementing Authorization and Authentication Profile Caching as a Failover Mechanism

The following configuration example shows how to:

- Create a cache profile group admin_users that contains all the administrators on the network so that if the RADIUS or TACACS+ server should become unavailable the administrators can still access the network.
- Create a cache profile group abc_users that contains all the *ABC* company users on the network so that if the RADIUS or TACACS+ server should become unavailable, these users will be authorized to use the network.

- Activate the new caching rules for each profile group on a RADIUS server.
- Add the new cache profile group in the authentication and authorization method list and change the method order so that the cache profile group is queried last.

```
configure terminal
 aaa new-model
 ! Define aaa cache profile groups and the rules for what information is saved to cache.
 aaa cache profile admin_users
 profile admin1
 profile admin2
 profile admin3
 exit
 aaa cache profile abcusers
 profile .*@example.com only no-auth
 exit
 ! Define server groups that use the cache information in each cache profile group.
 aaa group server tacacs+ admins@companyname.com
 server 10.1.1.1
 server 10.20.1.1
 cache authentication profile admin_users
 cache authorization profile admin_users
 exit
 aaa group server radius abcusers@example.com
 server 172.16.1.1
 server 172.20.1.1
 cache authentication profile abcusers
 cache authorization profile abcusers
 exit
 ! Update authentication and authorization method lists to specify how cache is used.
 aaa authentication login default cache admins@companyname.com group admins@companyname.com

 aaa authorization exec default cache admins@companyname.com group admins@companyname.com
 aaa authorization network default group abcusers@example.com cache abcusers@example.com
 end
```



CHAPTER 82

Enhanced IPv6 Neighbor Discovery Cache Management

Prerequisites for AAA Dead-Server Detection

- You must have access to a RADIUS server.
- You should be familiar with configuring a RADIUS server.
- You should be familiar with configuring authentication, authorization, and accounting (AAA).
- Before a server can be marked as dead, you must first configure the **radius-server deadtime** command. If this command is not configured, even if the criteria are met for the server to be marked as dead, the server state will be in the up state.

Restrictions for AAA Dead-Server Detection

- Original transmissions are not counted in the number of consecutive timeouts that must occur on the device before the server is marked as dead; only the number of retransmissions are counted.

Information About AAA Dead-Server Detection

This section provides information about the AAA Dead-Server Detection feature.

Criteria for Marking a RADIUS Server As Dead

The AAA Dead-Server Detection feature allows you to determine the criteria that are used to mark a RADIUS server as dead. That is, you can configure the minimum amount of time, in seconds, that must elapse from the time that the device last received a valid packet from the RADIUS server to the time the server is marked as dead. If a packet has not been received since the device booted, and there is a timeout, the time criterion will be treated as though it has been met.

In addition, you can configure the number of consecutive timeouts that must occur on the device before the RADIUS server is marked as dead. If the server performs both authentication and accounting, both types of packets are included in the number. Improperly constructed packets are counted as though they are timeouts. Only retransmissions are counted, not the initial transmission. (Each timeout causes one retransmission to be sent.)

Note Both the time criterion and the tries criterion must be met for the server to be marked as dead.

The RADIUS dead-server detection configuration will result in the prompt detection of RADIUS servers that have stopped responding. This configuration will also result in the avoidance of servers being improperly marked as dead when they are swamped (responding slowly) and the avoidance of the state of servers being rapidly changed from dead to live to dead again. This prompt detection of nonresponding RADIUS servers and the avoidance of swamped and dead-to-live-to-dead-again servers will result in less deadtime and quicker packet processing.

Each AAA RADIUS global and server groups can have its own deadtime configured. The deadtime configured on the server group takes precedence over the global deadtime configuration. When a deadtime is configured on any AAA RADIUS server group, it clears the existing dead timer on all global server groups that are marked as dead, and not just the specified server group.

How to Configure AAA Dead-Server Detection

This section describes how to configure AAA dead-server detection.

Configuring AAA Dead-Server Detection

To configure AAA Dead-Server Detection, perform the following steps.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables the AAA access control model.
Step 4	**radius-server deadtime** *minutes* **Example:** `Device(config)# radius-server deadtime 5`	Improves RADIUS response times when some servers might be unavailable and causes the unavailable servers to be skipped immediately.
Step 5	**radius-server dead-criteria** [**time** *seconds*] [**tries** *number-of-tries*] **Example:** `Device(config)# radius-server dead-criteria time 5 tries 4`	Forces one or both of the criteria, used to mark a RADIUS server as dead, to be the indicated constant.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your configuration. After you have configured AAA Dead-Server Detection, you should verify your configuration using this command. This verification is especially important if you have used the **no** form of the **radius-server dead-criteria** command. The output of this command must show the same values in the **Dead Criteria Details** field that you configured using the **radius-server dead-criteria** command.

Verifying AAA Dead-Server Detection

To verify your AAA Dead-Server Detection configuration, perform the following steps. The **show** and **debug** commands may be used in any order.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**debug aaa dead-criteria transactions** **Example:**	Displays AAA dead-criteria transaction values.

	Command or Action	Purpose
	`Device# debug aaa dead-criteria transactions`	
Step 3	**show aaa dead-criteria** **Example:** `Device# show aaa dead-criteria`	Displays dead-criteria information for a AAA server.
Step 4	**show aaa servers** [**private** | **public**] **Example:** `Device# show aaa server private`	Displays the status and number of packets that are sent to and received from all public and private authentication, authorization, and accounting (AAA) RADIUS servers. • The **private** keyword optionally displays the AAA servers only. • The **public** keyword optionally displays the AAA servers only.

Configuration Examples for AAA Dead-Server Detection

The following sections show configuration examples of AAA dead-server detection:

Example: Configuring AAA Dead-Server Detection

The following example shows that the device will be considered dead after 5 seconds and four tries:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# radius-server deadtime 5
Device(config)# radius-server dead-criteria time 5 tries 4
```

The following output example shows dead-criteria transaction information for a particular server group:

```
Device> enable
Device# debug aaa dead-criteria transactions

AAA Transaction debugs debugging is on
*Nov 14 23:44:17.403: AAA/SG/TRANSAC: Computed Retransmit Tries: 22, Current Max Tries: 22
*Nov 14 23:44:17.403: AAA/SG/TRANSAC: Computed Dead Detect Interval: 25s, Current Max
Interval: 25s
*Nov 14 23:44:17.403: AAA/SG/TRANSAC: Estimated Outstanding Transactions: 6, Current Max
Transactions: 6
```

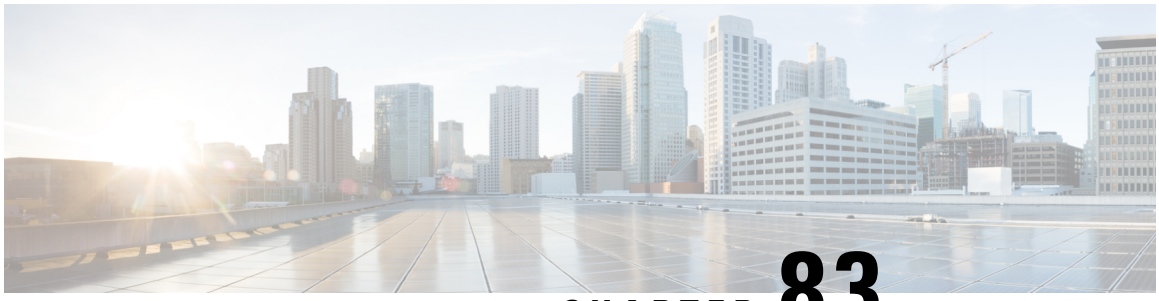
The following output example shows that dead-server-detection information has been requested for a RADIUS server at the IP address 192.0.2.1:

```
Device> enable
Device# show aaa dead-criteria radius 192.0.2.1 radius

RADIUS Server Dead Criteria:
============================
Server Details:
    Address : 192.0.2.1
```

```
    Auth Port : 1645
    Acct Port : 1646
Server Group : radius
Dead Criteria Details:
    Configured Retransmits : 62
    Configured Timeout : 27
    Estimated Outstanding Transactions: 5
    Dead Detect Time : 25s
    Computed Retransmit Tries: 22
    Statistics Gathered Since Last Successful Transaction
=====================================================
Max Computed Outstanding Transactions: 5
Max Computed Dead Detect Time: 25s
Max Computed Retransmits : 22
```

The following example shows that dead-criteria-detection information has been requested for a RADIUS server named ISE:

```
Device# show aaa dead-criteria radius server-name ISE

RADIUS Server Dead Criteria:
============================
Server Details:
    Address   : 192.0.2.2
    Auth Port : 1645
    Acct Port : 1646
Server Group  : radius
VRF           : Mgmt-vrf
Dead Criteria Details:
    Configured Retransmits   : 3
    Configured Timeout       : 5
    Estimated Outstanding Access Transactions: 0
    Estimated Outstanding Accounting Transactions: 0
    Dead Detect Time         : 5s
    Computed Retransmit Tries: 4
Statistics Gathered Since Last Successful Transaction
=====================================================
    Max Computed Outstanding Transactions: 1
    Max Computed Dead Detect Time: 10s
    Max Computed Retransmits : 10
```



CHAPTER 83

Configuring TACACS+

Prerequisites for TACACS+

The following are the prerequisites for set up and configuration of switch access with TACACS+ (must be performed in the order presented):

1. Configure the switches with the TACACS+ server addresses.
2. Set an authentication key.
3. Configure the key from Step 2 on the TACACS+ servers.
4. Enable authentication, authorization, and accounting (AAA).
5. Create a login authentication method list.
6. Apply the list to the terminal lines.
7. Create an authorization and accounting method list.

The following are the prerequisites for controlling switch access with TACACS+:

- You must have access to a configured TACACS+ server to configure TACACS+ features on your switch. Also, you must have access to TACACS+ services maintained in a database on a TACACS+ daemon typically running on a LINUX or Windows workstation.
- You need a system running the TACACS+ daemon software to use TACACS+ on your switch.
- To use TACACS+, it must be enabled.
- Authorization must be enabled on the switch to be used.
- Users must first successfully complete TACACS+ authentication before proceeding to TACACS+ authorization.
- To use any of the AAA commands listed in this section or elsewhere, you must first enable AAA with the **aaa new-model** command.

- At a minimum, you must identify the host or hosts maintaining the TACACS+ daemon and define the method lists for TACACS+ authentication. You can optionally define method lists for TACACS+ authorization and accounting.
- The method list defines the types of authentication to be performed and the sequence in which they are performed; it must be applied to a specific port before any of the defined authentication methods are performed. The only exception is the default method list (which, by coincidence, is named *default*). The default method list is automatically applied to all ports except those that have a named method list explicitly defined. A defined method list overrides the default method list.
- Use TACACS+ for privileged EXEC access authorization if authentication was performed by using TACACS+.
- Use the local database if authentication was not performed by using TACACS+.

Information About TACACS+

TACACS+ and Switch Access

This section describes TACACS+. TACACS+ provides detailed accounting information and flexible administrative control over the authentication and authorization processes. It is facilitated through authentication, authorization, accounting (AAA) and can be enabled only through AAA commands.

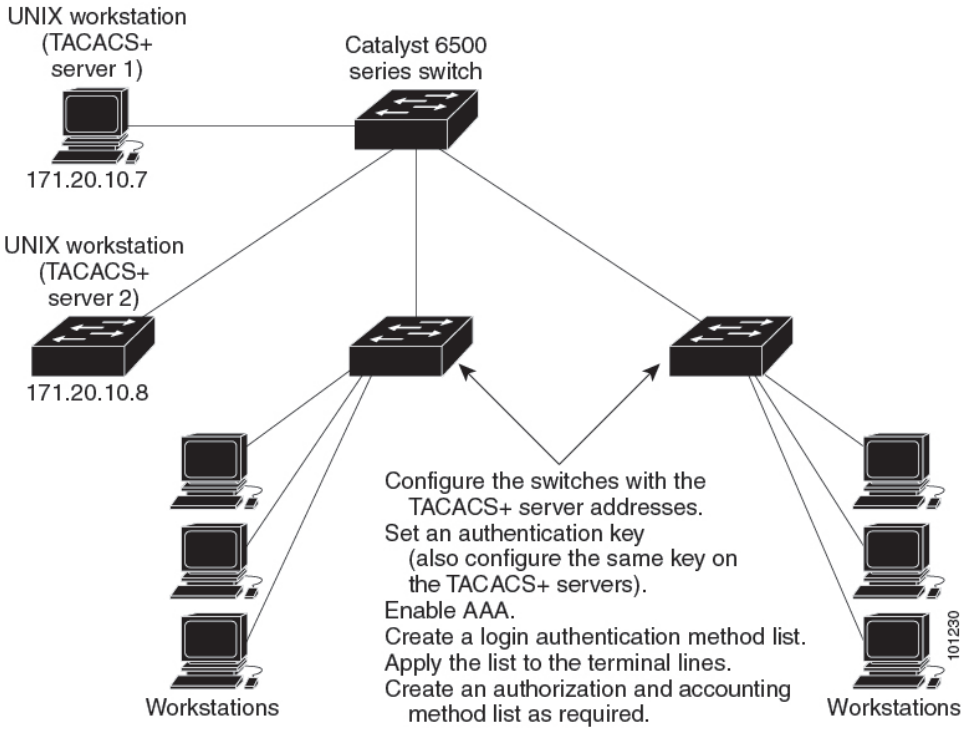
TACACS+ Overview

TACACS+ is a security application that provides centralized validation of users attempting to gain access to your switch.

TACACS+ provides for separate and modular authentication, authorization, and accounting facilities. TACACS+ allows for a single access control server (the TACACS+ daemon) to provide each service—authentication, authorization, and accounting—independently. Each service can be tied into its own database to take advantage of other services available on that server or on the network, depending on the capabilities of the daemon.

The goal of TACACS+ is to provide a method for managing multiple network access points from a single management service. Your switch can be a network access server along with other Cisco routers and access servers.

Figure 86: Network Configuration of Typical TACACS+



TACACS+, administered through the AAA security services, can provide these services:

- Authentication: Provides complete control of authentication through login and password dialog, challenge and response, and messaging support.

  The authentication facility can conduct a dialog with the user (for example, after a username and password are provided, to challenge a user with several questions, such as home address, mother's maiden name, service type, and social security number). The TACACS+ authentication service can also send messages to user screens. For example, a message could notify users that their passwords must be changed because of the company's password aging policy.

- Authorization: Provides fine-grained control over user capabilities for the duration of the user's session, including but not limited to setting autocommands, access control, session duration, or protocol support. You can also enforce restrictions on what commands a user can execute with the TACACS+ authorization feature.

- Accounting: Collects and sends information used for billing, auditing, and reporting to the TACACS+ daemon. Network managers can use the accounting facility to track user activity for a security audit or to provide information for user billing. Accounting records include user identities, start and stop times, executed commands, number of packets, and number of bytes.

The TACACS+ protocol provides authentication between the switch and the TACACS+ daemon, and it ensures confidentiality because all protocol exchanges between the switch and the TACACS+ daemon are encrypted.

TACACS+ Operation

When a user attempts a simple ASCII login by authenticating to a switch using TACACS+, this process occurs:

1. When the connection is established, the switch contacts the TACACS+ daemon to obtain a username prompt to show to the user. The user enters a username, and the switch then contacts the TACACS+ daemon to obtain a password prompt. The switch displays the password prompt to the user, the user enters a password, and the password is then sent to the TACACS+ daemon.

   TACACS+ allows a dialog between the daemon and the user until the daemon receives enough information to authenticate the user. The daemon prompts for a username and password combination, but can include other items, such as the user's mother's maiden name.

2. The switch eventually receives one of these responses from the TACACS+ daemon:

   - ACCEPT: The user is authenticated and service can begin. If the switch is configured to require authorization, authorization begins at this time.
   - REJECT: The user is not authenticated. The user can be denied access or is prompted to retry the login sequence, depending on the TACACS+ daemon.
   - ERROR: An error occurred at some time during authentication with the daemon or in the network connection between the daemon and the switch. If an ERROR response is received, the switch typically tries to use an alternative method for authenticating the user.
   - CONTINUE: The user is prompted for additional authentication information.

   After authentication, the user undergoes an additional authorization phase if authorization has been enabled on the switch. Users must first successfully complete TACACS+ authentication before proceeding to TACACS+ authorization.

3. If TACACS+ authorization is required, the TACACS+ daemon is again contacted, and it returns an ACCEPT or REJECT authorization response. If an ACCEPT response is returned, the response contains data in the form of attributes that direct the EXEC or NETWORK session for that user and the services that the user can access:

   - Telnet, Secure Shell (SSH), rlogin, or privileged EXEC services
   - Connection parameters, including the host or client IP address, access list, and user timeouts

Method List

A method list defines the sequence and methods to be used to authenticate, to authorize, or to keep accounts on a user. You can use method lists to designate one or more security protocols to be used, thus ensuring a backup system if the initial method fails. The software uses the first method listed to authenticate, to authorize, or to keep accounts on users; if that method does not respond, the software selects the next method in the list. This process continues until there is successful communication with a listed method or the method list is exhausted.

TACACS+ Configuration Options

You can configure the switch to use a single server or AAA server groups to group existing server hosts for authentication. You can group servers to select a subset of the configured server hosts and use them for a

particular service. The server group is used with a global server-host list and contains the list of IP addresses of the selected server hosts.

TACACS+ Login Authentication

A method list describes the sequence and authentication methods to be queried to authenticate a user. You can designate one or more security protocols to be used for authentication, thus ensuring a backup system for authentication in case the initial method fails. The software uses the first method listed to authenticate users; if that method fails to respond, the software selects the next authentication method in the method list. This process continues until there is successful communication with a listed authentication method or until all defined methods are exhausted. If authentication fails at any point in this cycle—meaning that the security server or local username database responds by denying the user access—the authentication process stops, and no other authentication methods are attempted.

TACACS+ Authorization for Privileged EXEC Access and Network Services

AAA authorization limits the services available to a user. When AAA authorization is enabled, the switch uses information retrieved from the user's profile, which is located either in the local user database or on the security server, to configure the user's session. The user is granted access to a requested service only if the information in the user profile allows it.

TACACS+ Accounting

The AAA accounting feature tracks the services that users are accessing and the amount of network resources that they are consuming. When AAA accounting is enabled, the switch reports user activity to the TACACS+ security server in the form of accounting records. Each accounting record contains accounting attribute-value (AV) pairs and is stored on the security server. This data can then be analyzed for network management, client billing, or auditing.

Default TACACS+ Configuration

TACACS+ and AAA are disabled by default.

To prevent a lapse in security, you cannot configure TACACS+ through a network management application. When enabled, TACACS+ can authenticate users accessing the switch through the CLI.

Note Although TACACS+ configuration is performed through the CLI, the TACACS+ server authenticates HTTP connections that have been configured with a privilege level of 15.

How to Configure TACACS+

This section describes how to configure your switch to support TACACS+.

Identifying the TACACS+ Server Host and Setting the Authentication Key

Follow these steps to identify the TACACS+ server host and set the authentication key:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**tacacs server** *server-name* **Example:** `Device(config)# tacacs server yourserver`	Identifies the IP host or hosts maintaining a TACACS+ server. Enter this command multiple times to create a list of preferred hosts. The software searches for hosts in the order in which you specify them. For *server-name*, specify the server name.
Step 4	**address** {**ipv4** | **ipv6**} *ip address* **Example:** `Device(config-server-tacacs)# address ipv4 10.0.1.12`	Configures the IP address for the TACACS server.
Step 5	**key** [*encryption-type*] [*key-string*] **Example:** `Device(config-server-tacacs)# key 0 auth-key`	Sets the authentication encryption key used for all TACACS+ communications between the access server and the TACACS+ daemon. This encryption key must match the key used on the TACACS+ daemon. *encryption-type* is optional, and if nothing is specified it is considered as clear text. Enter **0** to specify that an unencrypted key will follow. Enter **6** to specify that an encrypted key will follow. Enter **7** to specify that a hidden key will follow.
Step 6	**exit** **Example:** `Device(config-server-tacacs)# exit`	Exits the TACACS server mode and enters the global configuration mode.
Step 7	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.

	Command or Action	Purpose
Step 8	**aaa group server tacacs+** *group-name* **Example:** `Device(config)# aaa group server tacacs+ your_server_group`	(Optional) Defines the AAA server-group with a group name, and enters server group configuration mode.
Step 9	**server name** *server-name* **Example:** `Device(config-sg-tacacs)# server name yourserver`	(Optional) Associates a particular TACACS+ server with the defined server group. Repeat this step for each TACACS+ server in the AAA server group. Each server in the group must be previously defined in Step 3.
Step 10	**end** **Example:** `Device(config-sg-tacacs)# end`	Exits server group configuration mode and returns to privileged EXEC mode.

Configuring TACACS+ Login Authentication

Follow these steps to configure TACACS+ login authentication:

Before you begin

To configure AAA authentication, you define a named list of authentication methods and then apply that list to various ports.

Note To secure the device for HTTP access by using AAA methods, you must configure the device with the **ip http authentication aaa** global configuration command. Configuring AAA authentication does not secure the device for HTTP access by using AAA methods.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:**	Enables AAA.

	Command or Action	Purpose
	`Device(config)# aaa new-model`	
Step 4	**aaa authentication login** {**default** | *list-name*} *method1* [*method2...*] **Example:** `Device(config)# aaa authentication login default tacacs+ local`	Creates a login authentication method list. • To create a default list that is used when a named list is *not* specified in the **login authentication** command, use the **default** keyword followed by the methods that are to be used in default situations. The default method list is automatically applied to all ports. • For *list-name*, specify a character string to name the list you are creating. • For *method1...*, specify the actual method the authentication algorithm tries. The additional methods of authentication are used only if the previous method returns an error, not if it fails. Select one of these methods: • *enable*: Use the enable password for authentication. Before you can use this authentication method, you must define an enable password by using the **enable** *password* global configuration command. • *group tacacs+*: Uses TACACS+ authentication. Before you can use this authentication method, you must configure the TACACS+ server. • *line* : Use the line password for authentication. Before you can use this authentication method, you must define a line password. Use the **password** *password* line configuration command. • *local*: Use the local username database for authentication. You must enter username information in the database. Use the **username** *password* global configuration command. • *local-case*: Use a case-sensitive local username database for authentication. You must enter username information in the database by using the **username** *name* **password** global configuration command. • *none*: Do not use any authentication for login.

	Command or Action	Purpose
Step 5	**line** [**console** | **tty** | **vty**] *line-number* [*ending-line-number*] **Example:** `Device(config)# line 2 4`	Enters line configuration mode, and configures the lines to which you want to apply the authentication list.
Step 6	**login authentication** {**default** | *list-name*} **Example:** `Device(config-line)# login authentication default`	Applies the authentication list to a line or set of lines. • If you specify **default**, use the default list created with the **aaa authentication login** command. • For *list-name*, specify the list created with the **aaa authentication login** command.
Step 7	**end** **Example:** `Device(config-line)# end`	Exits line configuration mode and returns to privileged EXEC mode.

Configuring TACACS+ Authorization for Privileged EXEC Access and Network Services

You can use the **aaa authorization** global configuration command with the **tacacs+** keyword to set parameters that restrict a user's network access to privileged EXEC mode.

Note Authorization is bypassed for authenticated users who log in through the CLI even if authorization has been configured.

Follow these steps to specify TACACS+ authorization for privileged EXEC access and network services:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**aaa authorization network***authorization-list* **tacacs+** **Example:** `Device(config)# aaa authorization network list1 tacacs+`	Configures the switch for user TACACS+ authorization for all network-related service requests.
Step 4	**aaa authorization exec** *default***tacacs+** **Example:** `Device(config)# aaa authorization exec default tacacs+`	Configures the switch for user TACACS+ authorization if the user has privileged EXEC access. The **exec** keyword might return user profile information (such as **autocommand** information).
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Starting TACACS+ Accounting

Follow these steps to start TACACS+ Accounting:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa accounting network** *authorization-list* **start-stop tacacs+** **Example:** `Device(config)# aaa accounting network list1 start-stop tacacs+`	Enables TACACS+ accounting for all network-related service requests.
Step 4	**aaa accounting exec** *default* **start-stop tacacs+** **Example:** `Device(config)# aaa accounting exec default start-stop tacacs+`	Enables TACACS+ accounting to send a start-record accounting notice at the beginning of a privileged EXEC process and a stop-record at the end.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

What to do next

To establish a session with a device if the AAA server is unreachable, use the **aaa accounting system guarantee-first** command. It guarantees system accounting as the first record, which is the default condition. In some situations, users might be prevented from starting a session on the console or terminal connection until after the system reloads, which can take more than 3 minutes.

To establish a console or Telnet session with the router if the AAA server is unreachable when the router reloads, use the **no aaa accounting system guarantee-first** command.

Establishing a Session with a Device if the AAA Server is Unreachable

To establishing a session with a device if the AAA server is unreachable, use the **aaa accounting system guarantee-first** command. It guarantees system accounting as the first record, which is the default condition. In some situations, users might be prevented from starting a session on the console or terminal connection until after the system reloads, which can take more than 3 minutes.

To establish a console or Telnet session with the device if the AAA server is unreachable when the device reloads, use the **no aaa accounting system guarantee-first** command.

Configuring TACACS Source-Interface Under a TACACS Server-Group

The TACACS source-interface can be configured under a TACACS server-group in either of the following methods:

- Configure a TACACS source-interface under the TACACS server-group using the **ip tacacs source-interface** *interface-name* command.
- Configure a VRF using the **vrf** *vrf-name* command under the TACACS server-group, and then associate the configured VRF globally to a source-interface using the **ip tacacs source interface** *interface-name* **vrf** *vrf-name* command.

Priority will be given to the source-interface under the server-group configuration in case both methods are configured.

To configure TACACS source-interface under a TACACS server-group, perform the following:

Before you begin

You must configure a VRF routing table and associate VRF to an interface

Procedure

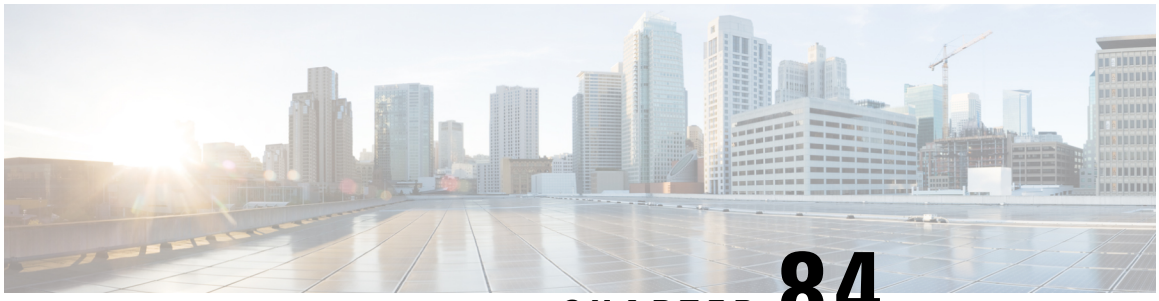
	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	{ **ip** | **ipv6** }**tacacs source-interface** *interface-number* **vrf** *vrf-name* **Example:** `Device(config)# ip tacacs source-interface GigabitEthernet1/1 vrf vrf17`	Forces TACACS to use the IP address of a specified interface for all outgoing TACACS packets, and enables the specification on a per-VRF basis. • *interface-name*: Specifies the name of the interface that TACACS+ uses for all of its outgoing packets. • **vrf** *vrf-name*: Specifies the per-VRF configuration.
Step 4	**aaa group server tacacs** *group_name* **Example:** `Device(config-sg-tacacs+)# aa group server tacacs rad-grp`	Groups different TACACS server hosts into distinct lists and distinct methods and enters server-group configuration mode.
Step 5	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-sg-tacacs+)# ip vrf forwarding vrf17`	(Optional) Configures a VRF for the interface.
Step 6	{ **ip** | **ipv6** } **tacacs source-interface** *interface-number* **Example:** `Device(config-sg-tacacs+)# ip tacacs source-interface loopback0`	(Optional) Forces TACACS+ to use the IP address of a specified interface for all outgoing TACACS packets from the TACACS+ group server. *interface-name*: Specifies the name of the interface that TACACS uses for all of its outgoing packets.
Step 7	**end** **Example:** `Device(config-sg-tacacs+)# end`	Returns to privileged EXEC mode.

Monitoring TACACS+

Table 91: Commands for Displaying TACACS+ Information

Command	Purpose
show tacacs	Displays TACACS+ server statistics.



CHAPTER 84

Configuring RADIUS

Prerequisites for Configuring RADIUS

This section lists the prerequisites for controlling device access with RADIUS.

General:

- RADIUS and Authentication, Authorization, and Accounting (AAA) must be enabled to use any of the configuration commands in this chapter.
- RADIUS is facilitated through AAA and can be enabled only through AAA commands.
- Use the **aaa new-model** global configuration command to enable AAA.
- Use the **aaa authentication** global configuration command to define method lists for RADIUS authentication.
- Use **line** and **interface** commands to enable the defined method lists to be used.
- At a minimum, you must identify the host or hosts that run the RADIUS server software and define the method lists for RADIUS authentication. You can optionally define method lists for RADIUS authorization and accounting.
- You should have access to and should configure a RADIUS server before configuring RADIUS features on your device.
- The RADIUS host is normally a multiuser system running RADIUS server software from Cisco (Cisco Secure Access Control Server Version 3.0), Livingston, Merit, Microsoft, or another software provider. For more information, see the RADIUS server documentation.
- To use the Change-of-Authorization (CoA) interface, a session must already exist on the switch. CoA can be used to identify a session and enforce a disconnect request. The update affects only the specified session.

RADIUS operation:

- Users must first successfully complete RADIUS authentication before proceeding to RADIUS authorization, if it is enabled.
- For RADIUS over IPv6 configurations, users must enable IPv6 unicast routing by enabling the **ipv6 unicast-routing** command.

Restrictions for Configuring RADIUS

General:

- To prevent a lapse in security, you cannot configure RADIUS through a network management application.

RADIUS is not suitable in the following network security situations:

- Multiprotocol access environments. RADIUS does not support AppleTalk Remote Access (ARA), NetBIOS Frame Control Protocol (NBFCP), NetWare Asynchronous Services Interface (NASI), or X.25 PAD connections.
- Switch-to-switch or router-to-router situations. RADIUS does not provide two-way authentication. RADIUS can be used to authenticate from one device to a non-Cisco device if the non-Cisco device requires authentication.
- Networks using a variety of services. RADIUS generally binds a user to one service model.

DSCP marking support for RADIUS packets:

- DSCP marking for authentication and accounting is not supported for private servers, fully qualified domain name (FQDN) servers and radsec servers.
- In the case of wired IEEE 802.1x authentication, when source port extension is not enabled, the default ports are in use. The DSCP marking is set to the default ports and all the requests will be marked with the same DSCP value.
- DSCP marking is not supported in the case of wireless IEEE 802.1x authentication, where the source port extension is enabled by default.

Information about RADIUS

RADIUS and Switch Access

This section describes how to enable and configure RADIUS. RADIUS provides detailed accounting information and flexible administrative control over the authentication and authorization processes.

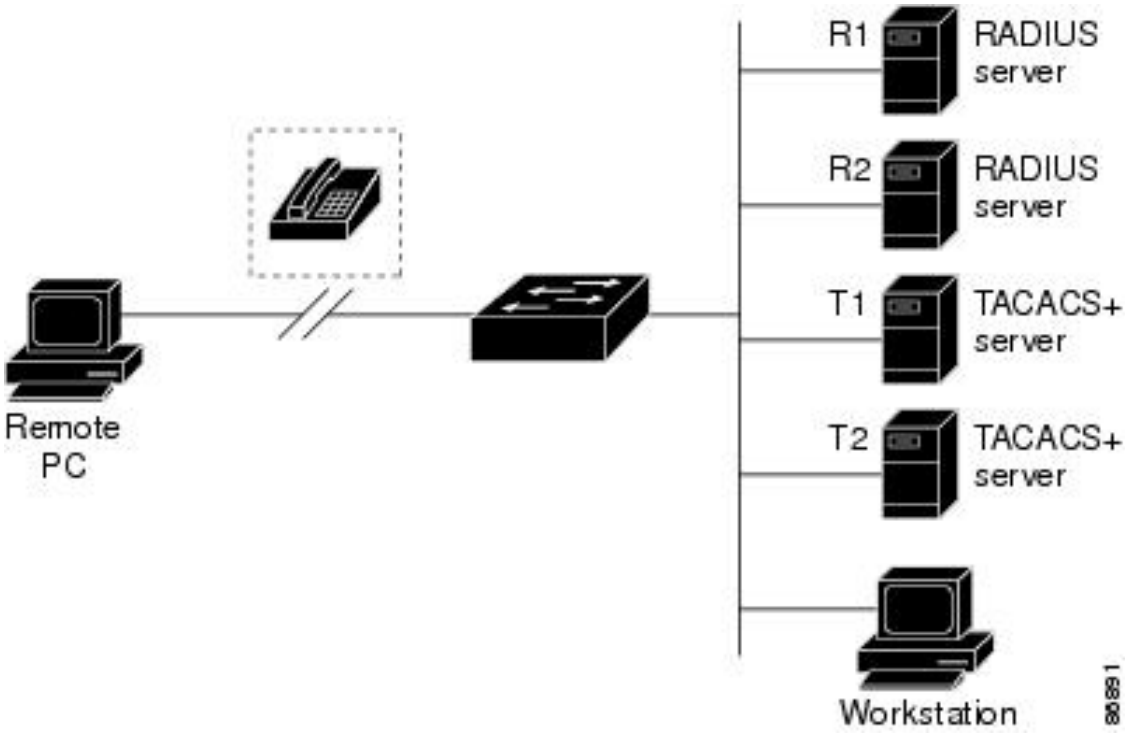
RADIUS Overview

RADIUS is a distributed client/server system that secures networks against unauthorized access. RADIUS clients run on supported Cisco devices. Clients send authentication requests to a central RADIUS server, which contains all user authentication and network service access information.

Use RADIUS in these network environments that require access security:

- Networks with multiple-vendor access servers, each supporting RADIUS. For example, access servers from several vendors use a single RADIUS server-based security database. In an IP-based network with multiple vendors' access servers, dial-in users are authenticated through a RADIUS server that has been customized to work with the Kerberos security system.
- Turnkey network security environments in which applications support the RADIUS protocol, such as in an access environment that uses a *smart card* access control system.
- Networks already using RADIUS. You can add a Cisco device containing a RADIUS client to the network. This might be the first step when you make a transition to a TACACS+ server. See the illustration: Transitioning from RADIUS to TACACS+ Services below.

Figure 87: Transitioning from RADIUS to TACACS+ Services



- Network in which the user must only access a single service. Using RADIUS, you can control user access to a single host, to a single utility such as Telnet, or to the network through a protocol such as IEEE 802.1x. For more information about this protocol, see the chapter *Configuring IEEE 802.1x Port-Based Authentication*.
- Networks that require resource accounting. You can use RADIUS accounting independently of RADIUS authentication or authorization. The RADIUS accounting functions allow data to be sent at the start and end of services, showing the amount of resources (such as time, packets, bytes, and so forth) used during the session. An Internet service provider might use a freeware-based version of RADIUS access control and accounting software to meet special security and billing needs.

RADIUS Operation

When a user attempts to log in and authenticate to a device that is access controlled by a RADIUS server, these events occur:

1. The user is prompted to enter a username and password.
2. The username and encrypted password are sent over the network to the RADIUS server.
3. The user receives one of the following responses from the RADIUS server:
    - ACCEPT: The user is authenticated.
    - REJECT: The user is either not authenticated and is prompted to re-enter the username and password, or access is denied.
    - CHALLENGE: A challenge requires additional data from the user.
    - CHALLENGE PASSWORD: A response requests the user to select a new password.

    The ACCEPT or REJECT response is bundled with additional data that is used for privileged EXEC or network authorization. The additional data included with the ACCEPT or REJECT packets includes these items:

- Telnet, SSH, rlogin, or privileged EXEC services
- Connection parameters, including the host or client IP address, access list, and user timeouts

RADIUS Change of Authorization

The RADIUS Change of Authorization (CoA) provides a mechanism to change the attributes of an authentication, authorization, and accounting (AAA) session after it is authenticated. When a policy changes for a user or user group in AAA, administrators can send RADIUS CoA packets from the AAA server such as a Cisco Secure Access Control Server (ACS) to reinitialize authentication and apply the new policy. This section provides an overview of the RADIUS interface including available primitives and how they are used during a CoA.

- Change-of-Authorization Requests
- CoA Request Response Code
- CoA Request Commands
- Session Reauthentication

A standard RADIUS interface is typically used in a pulled model where the request originates from a network attached device and the response come from the queried servers. Cisco devices support the RADIUS CoA extensions defined in RFC 5176 that are typically used in a pushed model and allow for the dynamic reconfiguring of sessions from external AAA or policy servers.

Cisco devices supports these per-session CoA requests:

- Session reauthentication
- Session termination
- Session termination with port shutdown

- Session termination with port bounce

This feature is integrated with Cisco Identity Services Engine (ISE).

The RADIUS interface is enabled by default on Cisco devices. However, some basic configuration is required for the following attributes:

- Security and Password—refer to the "Preventing Unauthorized Access to Your Switch" section in this guide.
- Accounting—refer to the "Starting RADIUS Accounting" section in the Configuring Switch-Based Authentication chapter in this guide.

Cisco IOS XE software supports the RADIUS CoA extensions defined in RFC 5176 that are typically used in a push model to allow the dynamic reconfiguring of sessions from external AAA or policy servers. Per-session CoA requests are supported for session identification, session termination, host reauthentication, port shutdown, and port bounce. This model comprises one request (CoA-Request) and two possible response codes:

- CoA acknowledgement (ACK) [CoA-ACK]
- CoA nonacknowledgement (NAK) [CoA-NAK]

The request is initiated from a CoA client (typically a AAA or policy server) and directed to the device that acts as a listener.

The table below shows the RADIUS CoA commands and vendor-specific attributes (VSAs) supported by Identity-Based Networking Services. All CoA commands must include the session identifier between the device and the CoA client.

Table 92: RADIUS CoA Commands Supported by Identity-Based Networking Services

CoA Command	Cisco VSA
Activate service	Cisco:Avpair="subscriber:command=activate-service" Cisco:Avpair="subscriber:service-name=<*service-name*>" Cisco:Avpair="subscriber:precedence=<*precedence-number*>" Cisco:Avpair="subscriber:activation-mode=replace-all"
Deactivate service	Cisco:Avpair="subscriber:command=deactivate-service" Cisco:Avpair="subscriber:service-name=<*service-name*>"
Bounce host port	Cisco:Avpair="subscriber:command=bounce-host-port"
Disable host port	Cisco:Avpair="subscriber:command=disable-host-port"
Session query	Cisco:Avpair="subscriber:command=session-query"
Session reauthenticate	Cisco:Avpair="subscriber:command=reauthenticate" Cisco:Avpair="subscriber:reauthenticate-type=last" or Cisco:Avpair="subscriber:reauthenticate-type=rerun"
Session terminate	This is a standard disconnect request and does not require a VSA.

CoA Command	Cisco VSA
Interface template	Cisco:AVpair="interface-template-name=<*interfacetemplate*>"

Change-of-Authorization Requests

Change of Authorization (CoA) requests, as described in RFC 5176, are used in a push model to allow for session identification, host reauthentication, and session termination. The model is comprised of one request (CoA-Request) and two possible response codes:

- CoA acknowledgment (ACK) [CoA-ACK]
- CoA non-acknowledgment (NAK) [CoA-NAK]

The request is initiated from a CoA client (typically a RADIUS or policy server) and directed to the switch that acts as a listener.

RFC 5176 Compliance

The Disconnect Request message, which is also referred to as Packet of Disconnect (POD), is supported by the switch for session termination.

This table shows the IETF attributes are supported for this feature.

Table 93: Supported IETF Attributes

Attribute Number	Attribute Name
24	State
31	Calling-Station-ID
44	Acct-Session-ID
80	Message-Authenticator
101	Error-Cause

This table shows the possible values for the Error-Cause attribute.

Table 94: Error-Cause Values

Value	Explanation
201	Residual Session Context Removed
202	Invalid EAP Packet (Ignored)
401	Unsupported Attribute
402	Missing Attribute
403	NAS Identification Mismatch

Value	Explanation
404	Invalid Request
405	Unsupported Service
406	Unsupported Extension
407	Invalid Attribute Value
501	Administratively Prohibited
502	Request Not Routable (Proxy)
503	Session Context Not Found
504	Session Context Not Removable
505	Other Proxy Processing Error
506	Resources Unavailable
507	Request Initiated
508	Multiple Session Selection Unsupported

CoA Request Response Code

The CoA Request response code can be used to convey a command to the switch.

The packet format for a CoA Request Response code as defined in RFC 5176 consists of the following fields: Code, Identifier, Length, Authenticator, and Attributes in the Type:Length:Value (TLV) format. The Attributes field is used to carry Cisco vendor-specific attributes (VSAs).

Session Identification

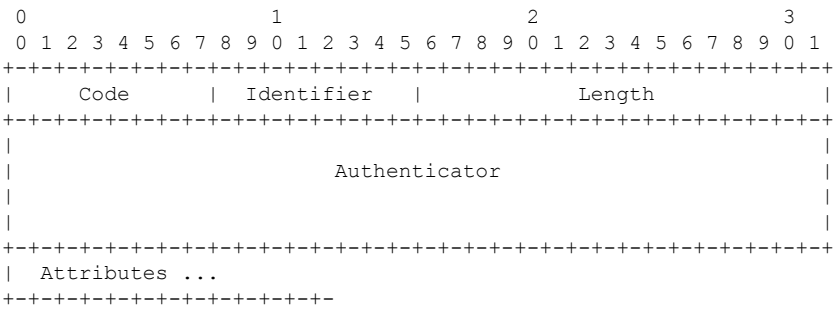
For disconnect and CoA requests targeted at a particular session, the switch locates the session based on one or more of the following attributes:

- Acct-Session-Id (IETF attribute #44)
- Audit-Session-Id (Cisco VSA)
- Calling-Station-Id (IETF attribute #31 which contains the host MAC address)
- IPv6 Attributes, which can be one of the following:
  - Framed-IPv6-Prefix (IETF attribute #97) and Framed-Interface-Id (IETF attribute #96), which together create a full IPv6 address per RFC 3162
  - Framed-IPv6-Address
- Plain IP Address (IETF attribute #8)

Unless all session identification attributes included in the CoA message match the session, the switch returns a Disconnect-NAK or CoA-NAK with the “Invalid Attribute Value” error-code attribute.

If more than one session identification attribute is included in the message, all the attributes must match the session or the switch returns a Disconnect- negative acknowledgment (NAK) or CoA-NAK with the error code “Invalid Attribute Value.”

The packet format for a CoA Request code as defined in RFC 5176 consists of the fields: Code, Identifier, Length, Authenticator, and Attributes in Type:Length:Value (TLV) format.



The attributes field is used to carry Cisco vendor-specific attributes (VSAs).

For CoA requests targeted at a particular enforcement policy, the device returns a CoA-NAK with the error code “Invalid Attribute Value” if any of the above session identification attributes are included in the message.

CoA ACK Response Code

If the authorization state is changed successfully, a positive acknowledgment (ACK) is sent. The attributes returned within CoA ACK will vary based on the CoA Request and are discussed in individual CoA Commands.

CoA NAK Response Code

A negative acknowledgment (NAK) indicates a failure to change the authorization state and can include attributes that indicate the reason for the failure. Use **show** commands to verify a successful CoA.

CoA Request Commands

Table 95: Supported CoA Commands

Command [7]	Cisco VSA
Reauthenticate host	Cisco:Avpair=“subscriber:command=reauthenticate”
Terminate session	This is a standard disconnect request that does not require a VSA.
Bounce host port	Cisco:Avpair=“subscriber:command=bounce-host-port”
Disable host port	Cisco:Avpair=“subscriber:command=disable-host-port”

[7] All CoA commands must include the session identifier between the device and the CoA client.

Session Reauthentication

The AAA server typically generates a session reauthentication request when a host with an unknown identity or posture joins the network and is associated with a restricted access authorization profile (such as a guest VLAN). A reauthentication request allows the host to be placed in the appropriate authorization group when its credentials are known.

To initiate session authentication, the AAA server sends a standard CoA-Request message which contains a Cisco VSA in this form: *Cisco:Avpair="subscriber:command=reauthenticate"* and one or more session identification attributes.

The current session state determines the switch response to the message. If the session is currently authenticated by IEEE 802.1x, the switch responds by sending an EAPoL (Extensible Authentication Protocol over Lan) -RequestId message to the server.

If the session is currently authenticated by MAC authentication bypass (MAB), the switch sends an access-request to the server, passing the same identity attributes used for the initial successful authentication.

If session authentication is in progress when the switch receives the command, the switch terminates the process, and restarts the authentication sequence, starting with the method configured to be attempted first.

If the session is not yet authorized, or is authorized via guest VLAN, or critical VLAN, or similar policies, the reauthentication message restarts the access control methods, beginning with the method configured to be attempted first. The current authorization of the session is maintained until the reauthentication leads to a different authorization result.

Session Termination

There are three types of CoA requests that can trigger session termination. A CoA Disconnect-Request terminates the session, without disabling the host port. This command causes re-initialization of the authenticator state machine for the specified host, but does not restrict that host access to the network.

To restrict a host's access to the network, use a CoA Request with the Cisco:Avpair="subscriber:command=disable-host-port" VSA. This command is useful when a host is known to be causing problems on the network, and you need to immediately block network access for the host. When you want to restore network access on the port, re-enable it using a non-RADIUS mechanism.

When a device with no supplicant, such as a printer, needs to acquire a new IP address (for example, after a VLAN change), terminate the session on the host port with port-bounce (temporarily disable and then re-enable the port).

CoA Disconnect-Request

This command is a standard Disconnect-Request. If the session cannot be located, the device returns a Disconnect-NAK message with the "Session Context Not Found" error-code attribute. If the session is located, the device terminates the session. After the session has been completely removed, the device returns a Disconnect-ACK.

If the device fails-over to a standby device before returning a Disconnect-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the session is not found following re-sending, a Disconnect-ACK is sent with the "Session Context Not Found" error-code attribute.

CoA Request: Disable Host Port

The RADIUS server CoA disable port command administratively shuts down the authentication port that is hosting a session, resulting in session termination. This command is useful when a host is known to cause problems on the network and network access needs to be immediately blocked for the host. To restore network

access on the port, reenable it using a non-RADIUS mechanism. This command is carried in a standard CoA-Request message that has this new vendor-specific attribute (VSA):

Cisco:Avpair="subscriber:command=disable-host-port"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the “Session Identification” section. If the session cannot be located, the device returns a CoA-NAK message with the “Session Context Not Found” error-code attribute. If the session is located, the device disables the hosting port and returns a CoA-ACK message.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation has completed, the operation is restarted on the new active device.

Note A Disconnect-Request failure following command re-sending could be the result of either a successful session termination before change-over (if the Disconnect-ACK was not sent) or a session termination by other means (for example, a link failure) that occurred after the original command was issued and before the standby device became active.

CoA Request: Bounce-Port

A RADIUS server CoA bounce port sent from a RADIUS server can cause a link flap on an authentication port, which triggers DHCP renegotiation from one or more hosts connected to this port. This incident can occur when there is a VLAN change and the endpoint is a device (such as a printer) that does not have a mechanism to detect a change on this authentication port. The CoA bounce port is carried in a standard CoA-Request message that contains the following VSA:

Cisco:Avpair="subscriber:command=bounce-host-port"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes. If the session cannot be located, the device returns a CoA-NAK message with the “Session Context Not Found” error-code attribute. If the session is located, the device disables the hosting port for a period of 10 seconds, re-enables it (port-bounce), and returns a CoA-ACK.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation has completed, the operation is re-started on the new active device.

Default RADIUS Configuration

RADIUS and AAA are disabled by default.

To prevent a lapse in security, you cannot configure RADIUS through a network management application. When enabled, RADIUS can authenticate users accessing the device through the CLI.

RADIUS Server Host

Device-to-RADIUS-server communication involves several components:

- Hostname or IP address
- Authentication destination port

- Accounting destination port
- Key string
- Timeout period
- Retransmission value

You identify RADIUS security servers by their hostname or IP address, hostname and specific UDP port numbers, or their IP address and specific UDP port numbers. The combination of the IP address and the UDP port number creates a unique identifier, allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. This unique identifier enables RADIUS requests to be sent to multiple UDP ports on a server at the same IP address.

If two different host entries on the same RADIUS server are configured for the same service—for example, accounting—the second host entry configured acts as a fail-over backup to the first one. Using this example, if the first host entry fails to provide accounting services, the %RADIUS-4-RADIUS_DEAD message appears, and then the device tries the second host entry configured on the same device for accounting services. (The RADIUS host entries are tried in the order that they are configured.)

A RADIUS server and the device use a shared secret text string to encrypt passwords and exchange responses. To configure RADIUS to use the AAA security commands, you must specify the host running the RADIUS server daemon and a secret text (key) string that it shares with the device.

The timeout, retransmission, and encryption key values can be configured globally for all RADIUS servers, on a per-server basis, or in some combination of global and per-server settings.

RADIUS Login Authentication

To configure AAA authentication, you define a named list of authentication methods and then apply that list to various ports. The method list defines the types of authentication to be performed and the sequence in which they are performed; it must be applied to a specific port before any of the defined authentication methods are performed. The only exception is the default method list. The default method list is automatically applied to all ports except those that have a named method list explicitly defined.

A method list describes the sequence and authentication methods to be queried to authenticate a user. You can designate one or more security protocols to be used for authentication, thus ensuring a backup system for authentication in case the initial method fails. The software uses the first method listed to authenticate users; if that method fails to respond, the software selects the next authentication method in the method list. This process continues until there is successful communication with a listed authentication method or until all defined methods are exhausted. If authentication fails at any point in this cycle—meaning that the security server or local username database responds by denying the user access—the authentication process stops, and no other authentication methods are attempted.

AAA Server Groups

You can configure the device to use AAA server groups to group existing server hosts for authentication. You select a subset of the configured server hosts and use them for a particular service. The server group is used with a global server-host list, which lists the IP addresses of the selected server hosts.

Server groups also can include multiple host entries for the same server if each entry has a unique identifier (the combination of the IP address and UDP port number), allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. This unique identifier enables RADIUS requests to be sent to different UDP ports on a server at the same IP address. If you configure two different host entries on

the same RADIUS server for the same service, (for example, accounting), the second configured host entry acts as a fail-over backup to the first one. If the first host entry fails to provide accounting services, the network access server tries the second host entry configured on the same device for accounting services. (The RADIUS host entries are tried in the order in which they are configured.)

AAA Authorization

AAA authorization limits the services available to a user. When AAA authorization is enabled, the device uses information retrieved from the user's profile, which is in the local user database or on the security server, to configure the user's session. The user is granted access to a requested service only if the information in the user profile allows it.

RADIUS Accounting

The AAA accounting feature tracks the services that users are using and the amount of network resources that they are consuming. When you enable AAA accounting, the device reports user activity to the RADIUS security server in the form of accounting records. Each accounting record contains accounting attribute-value (AV) pairs and is stored on the security server. You can then analyze the data for network management, client billing, or auditing.

Vendor-Specific RADIUS Attributes

The Internet Engineering Task Force (IETF) draft standard specifies a method for communicating vendor-specific information between the device and the RADIUS server by using the vendor-specific attribute (attribute 26). Vendor-specific attributes (VSAs) allow vendors to support their own extended attributes not suitable for general use. The Cisco RADIUS implementation supports one vendor-specific option by using the format recommended in the specification. Cisco's vendor-ID is 9, and the supported option has vendor-type 1, which is named *cisco-avpair*. The value is a string with this format:

```
protocol : attribute sep value *
```

Protocol is a value of the Cisco protocol attribute for a particular type of authorization. *Attribute* and *value* are an appropriate attributevalue (AV) pair defined in the Cisco TACACS+ specification, and *sep* is = for mandatory attributes and is * for optional attributes. The full set of features available for TACACS+ authorization can then be used for RADIUS.

For example, the following AV pair causes Cisco's "multiple named IP address pools" feature to be activated during IP authorization :

```
cisco-avpair= ″ip:addr-pool=first‟
```

If you insert an "*", the AV pair "ip:addr-pool=first" becomes optional. Note that any AV pair can be made optional:

```
cisco-avpair= ″ip:addr-pool*first‟
```

The following example shows how to cause a user logging in from a network access server to have immediate access to EXEC commands:

```
cisco-avpair= ″shell:priv-lvl=15‟
```

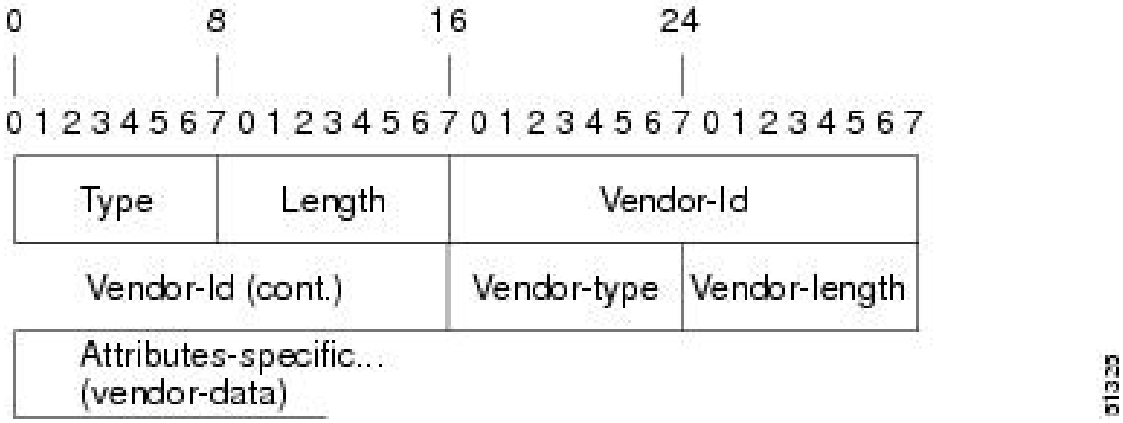
Other vendors have their own unique vendor-IDs, options, and associated VSAs. For more information about vendor-IDs and VSAs, see RFC 2138, "Remote Authentication Dial-In User Service (RADIUS)."

Attribute 26 contains the following three elements:

- Type
- Length
- String (also known as data)
  - Vendor-ID
  - Vendor-Type
  - Vendor-Length
  - Vendor-Data

The figure below shows the packet format for a VSA encapsulated "behind" attribute 26.

Figure 88: VSA Encapsulated Behind Attribute 26



Note It is up to the vendor to specify the format of their VSA. The Attribute-Specific field (also known as Vendor-Data) is dependent on the vendor's definition of that attribute.

The table below describes significant fields listed in the Vendor-Specific RADIUS IETF Attributes table (second table below), which lists supported vendor-specific RADIUS attributes (IETF attribute 26).

Table 96: Vendor-Specific Attributes Table Field Descriptions

Field	Description
Number	All attributes listed in the following table are extensions of IETF attribute 26.
Vendor-Specific Command Codes	A defined code used to identify a particular vendor. Code 9 defines Cisco VSAs, 311 defines Microsoft VSAs, and 529 defines Ascend VSAs.
Sub-Type Number	The attribute ID number. This number is much like the ID numbers of IETF attributes, except it is a "second layer" ID number encapsulated behind attribute 26.
Attribute	The ASCII string name of the attribute.
Description	Description of the attribute.

Table 97: Vendor-Specific RADIUS IETF Attributes

Number	Vendor-Specific Company Code	Sub-Type Number	Attribute	Description
VPDN Attributes				
26	9	1	l2tp-cm-local-window-size	Specifies the maximum receive window size for L2TP control messages. This value is advertised to the peer during tunnel establishment.
26	9	1	l2tp-drop-out-of-order	Respects sequence numbers on data packets by dropping those that are received out of order. This does not ensure that sequence numbers will be sent on data packets, just how to handle them if they are received.
26	9	1	l2tp-hello-interval	Specifies the number of seconds for the hello keepalive interval. Hello packets are sent when no data has been sent on a tunnel for the number of seconds configured here.
26	9	1	l2tp-hidden-avp	When enabled, sensitive AVPs in L2TP control messages are scrambled or hidden.
26	9	1	l2tp-nosession-timeout	Specifies the number of seconds that a tunnel will stay active with no sessions before timing out and shutting down.
26	9	1	tunnel-tos-reflect	Copies the IP ToS field from the IP header of each payload packet to the IP header of the tunnel packet for packets entering the tunnel at the LNS.
26	9	1	l2tp-tunnel-authen	If this attribute is set, it performs L2TP tunnel authentication.

Number	Vendor-Specific Company Code	Sub-Type Number	Attribute	Description
26	9	1	l2tp-tunnel-password	Shared secret used for L2TP tunnel authentication and AVP hiding.
26	9	1	l2tp-udp-checksum	This is an authorization attribute and defines whether L2TP should perform UDP checksums for data packets. Valid values are “yes” and “no.” The default is no.
Store and Forward Fax Attributes				
26	9	3	Fax-Account-Id-Origin	Indicates the account ID origin as defined by system administrator for the **mmoip aaa receive-id** or the **mmoip aaa send-id** commands.
26	9	4	Fax-Msg-Id=	Indicates a unique fax message identification number assigned by Store and Forward Fax.
26	9	5	Fax-Pages	Indicates the number of pages transmitted or received during this fax session. This page count includes cover pages.
26	9	6	Fax-Coverpage-Flag	Indicates whether or not a cover page was generated by the off-ramp gateway for this fax session. True indicates that a cover page was generated; false means that a cover page was not generated.

Number	Vendor-Specific Company Code	Sub-Type Number	Attribute	Description
26	9	7	Fax-Modem-Time	Indicates the amount of time in seconds the modem sent fax data (x) and the amount of time in seconds of the total fax session (y), which includes both fax-mail and PSTN time, in the form x/y. For example, 10/15 means that the transfer time took 10 seconds, and the total fax session took 15 seconds.
26	9	8	Fax-Connect-Speed	Indicates the modem speed at which this fax-mail was initially transmitted or received. Possible values are 1200, 4800, 9600, and 14400.
26	9	9	Fax-Recipient-Count	Indicates the number of recipients for this fax transmission. Until e-mail servers support Session mode, the number should be 1.
26	9	10	Fax-Process-Abort-Flag	Indicates that the fax session was terminated or successful. True means that the session was terminated; false means that the session was successful.
26	9	11	Fax-Dsn-Address	Indicates the address to which DSNs will be sent.
26	9	12	Fax-Dsn-Flag	Indicates whether or not DSN has been enabled. True indicates that DSN has been enabled; false means that DSN has not been enabled.
26	9	13	Fax-Mdn-Address	Indicates the address to which MDNs will be sent.

Number	Vendor-Specific Company Code	Sub-Type Number	Attribute	Description
26	9	14	Fax-Mdn-Flag	Indicates whether or not message delivery notification (MDN) has been enabled. True indicates that MDN had been enabled; false means that MDN had not been enabled.
26	9	15	Fax-Auth-Status	Indicates whether or not authentication for this fax session was successful. Possible values for this field are success, failed, bypassed, or unknown.
26	9	16	Email-Server-Address	Indicates the IP address of the e-mail server handling the on-ramp fax-mail message.
26	9	17	Email-Server-Ack-Flag	Indicates that the on-ramp gateway has received a positive acknowledgment from the e-mail server accepting the fax-mail message.
26	9	18	Gateway-Id	Indicates the name of the gateway that processed the fax session. The name appears in the following format: hostname.domain-name.
26	9	19	Call-Type	Describes the type of fax activity: fax receive or fax send.
26	9	20	Port-Used	Indicates the slot/port number of the Cisco AS5300 used to either transmit or receive this fax-mail.

Number	Vendor-Specific Company Code	Sub-Type Number	Attribute	Description
26	9	21	Abort-Cause	If the fax session terminates, indicates the system component that signaled the termination. Examples of system components that could trigger an termination are FAP (Fax Application Process), TIFF (the TIFF reader or the TIFF writer), fax-mail client, fax-mail server, ESMTP client, or ESMTP server.
H323 Attributes				
26	9	23	Remote-Gateway-ID (h323-remote-address)	Indicates the IP address of the remote gateway.
26	9	24	Connection-ID (h323-conf-id)	Identifies the conference ID.
26	9	25	Setup-Time (h323-setup-time)	Indicates the setup time for this connection in Coordinated Universal Time (UTC) formerly known as Greenwich Mean Time (GMT) and Zulu time.
26	9	26	Call-Origin (h323-call-origin)	Indicates the origin of the call relative to the gateway. Possible values are originating and terminating (answer).
26	9	27	Call-Type (h323-call-type)	Indicates call leg type. Possible values are **telephony** and **VoIP**.
26	9	28	Connect-Time (h323-connect-time)	Indicates the connection time for this call leg in UTC.
26	9	29	Disconnect-Time (h323-disconnect-time)	Indicates the time this call leg was disconnected in UTC.

Number	Vendor-Specific Company Code	Sub-Type Number	Attribute	Description
26	9	30	Disconnect-Cause (h323-disconnect-cause)	Specifies the reason a connection was taken offline per Q.931 specification.
26	9	31	Voice-Quality (h323-voice-quality)	Specifies the impairment factor (ICPIF) affecting voice quality for a call.
26	9	33	Gateway-ID (h323-gw-id)	Indicates the name of the underlying gateway.
Large Scale Dialout Attributes				
26	9	1	callback-dialstring	Defines a dialing string to be used for callback.
26	9	1	data-service	No description available.
26	9	1	dial-number	Defines the number to dial.
26	9	1	force-56	Determines whether the network access server uses only the 56 K portion of a channel, even when all 64 K appear to be available.
26	9	1	map-class	Allows the user profile to reference information configured in a map class of the same name on the network access server that dials out.
26	9	1	send-auth	Defines the protocol to use (PAP or CHAP) for username-password authentication following CLID authentication.

Number	Vendor-Specific Company Code	Sub-Type Number	Attribute	Description
26	9	1	remote-name	Provides the name of the remote host for use in large-scale dial-out. Dialer checks that the large-scale dial-out remote name matches the authenticated name, to protect against accidental user RADIUS misconfiguration. (For example, dialing a valid phone number but connecting to the wrong device.)
Miscellaneous Attributes				
26	9	2	Cisco-NAS-Port	Specifies additional vendor specific attribute (VSA) information for NAS-Port accounting. To specify additional NAS-Port information in the form an Attribute-Value Pair (AVPair) string, use the **radius-server vsa send** global configuration command. **Note** This VSA is typically used in Accounting, but may also be used in Authentication (Access-Request) packets.
26	9	1	min-links	Sets the minimum number of links for MLP.
26	9	1	proxyacl#<n>	Allows users to configure the downloadable user profiles (dynamic ACLs) by using the authentication proxy feature so that users can have the configured authorization to permit traffic going through the configured interfaces.

Number	Vendor-Specific Company Code	Sub-Type Number	Attribute	Description
26	9	1	spi	Carries the authentication information needed by the home agent to authenticate a mobile node during registration. The information is in the same syntax as the **ip mobile secure host <addr>** configuration command. Basically it contains the rest of the configuration command that follows that string, verbatim. It provides the Security Parameter Index (SPI), key, authentication algorithm, authentication mode, and replay protection timestamp range.

Vendor-Proprietary RADIUS Server Communication

Although an IETF draft standard for RADIUS specifies a method for communicating vendor-proprietary information between the device and the RADIUS server, some vendors have extended the RADIUS attribute set in a unique way. Cisco IOS XE software supports a subset of vendor-proprietary RADIUS attributes.

As mentioned earlier, to configure RADIUS (whether vendor-proprietary or IETF draft-compliant), you must specify the host running the RADIUS server daemon and the secret text string it shares with the device. You specify the RADIUS host and secret text string by using the **radius server** global configuration commands.

DSCP marking for RADIUS packets

Differentiated Services (DiffServ) is a model in which traffic is treated by intermediate systems with relative priorities based on the type of services (ToS) field. The six most significant bits of the DiffServ field is called as the Differentiated Services Code Point (DSCP). Cisco IOS XE software supports DSCP marking for RADIUS packets. DSCP marking enables faster authentication and accounting of RADIUS packets.

You can configure DSCP marking on the RADIUS server, server group and in global configuration mode. When DSCP marking configuration is applied on RADIUS server, server group and global configuration mode, the DSCP marking values entered on the RADIUS server is taken.

- If there is no DSCP marking configuration on the RADIUS server, the DSCP marking values configured on the server group is applied to the RADIUS packets.
- If there is no DSCP marking configuration on the RADIUS server, RADIUS server group, the DSCP marking values configured at the global configuration mode is applied to the RADIUS packets.

How to Configure RADIUS

Identifying the RADIUS Server Host

To apply these settings globally to all RADIUS servers communicating with the device, use the three unique global configuration commands: **radius-server timeout**, **radius-server retransmit**, and **key** *string*.

You can configure the device to use AAA server groups to group existing server hosts for authentication.

You also need to configure some settings on the RADIUS server. These settings include the IP address of the device and the key string to be shared by both the server and the device.

Follow these steps to configure per-server RADIUS server communication.

Before you begin

If you configure both global and per-server functions (timeout, retransmission, and key commands) on the device, the per-server timer, retransmission, and key value commands override global timer, retransmission, and key value commands.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**radius server** *server name* **Example:** `Device(config)# radius server rsim`	Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode.
Step 4	**address** {**ipv4** | **ipv6**} *ip address*{ **auth-port** *port number* | **acct-port** *port number*} **Example:** `Device(config-radius-server)# address ipv4 124.2.2.12 auth-port 1612`	(Optional) Specifies the RADIUS server parameters. For **auth-port** *port-number*, specify the UDP destination port for authentication requests. The default is 1645. The range is 0 to 65536. For **acct-port** *port-number*, specify the UDP destination port for authentication requests. The default is 1646.
Step 5	**key** *string* **Example:**	(Optional) For **key** *string*, specify the authentication and encryption key used between

	Command or Action	Purpose
	`Device(config-radius-server)# key rad123`	the device and the RADIUS daemon running on the RADIUS server. **Note** The key is a text string that must match the encryption key used on the RADIUS server. Always configure the key as the last item in the **radius server** command. Leading spaces are ignored, but spaces within and at the end of the key are used. If you use spaces in your key, do not enclose the key in quotation marks unless the quotation marks are part of the key.
Step 6	**retransmit** *value* **Example:** `Device(config-radius-server)# retransmit 10`	(Optional) Specifies the number of times a RADIUS request is resent when the server is not responding or responding slowly. The range is 1 to 100. This setting overrides the **radius-server retransmit** global configuration command setting.
Step 7	**timeout** *seconds* **Example:** `Device(config-radius-server)# timeout 60`	(Optional) Specifies the time interval that the device waits for the RADIUS server to reply before sending a request again. The range is 1 to 1000. This setting overrides the **radius-server timeout** global configuration command setting. **Note** We recommend that you configure timeout under the **radius-server timeout** command only and not under the **aaa group server radius** command.
Step 8	**end** **Example:** `Device(config-radius-server)# end`	Exits RADIUS server configuration mode and enters privileged EXEC mode.

Configuring RADIUS Login Authentication

Follow these steps to configure RADIUS login authentication:

Before you begin

To secure the device for HTTP access by using AAA methods, you must configur the **ip http authentication aaa** global configuration command. Configuring AAA authentication does not secure the device for HTTP access by using AAA methods.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**aaa authentication login** {**default** | *list-name*} *method1* [*method2*...] **Example:** `Device(config)# aaa authentication login default local`	Creates a login authentication method list. • To create a default list that is used when a named list is *not* specified in the **login authentication** command, use the **default** keyword followed by the methods that are to be used in default situations. The default method list is automatically applied to all ports. • For *list-name*, specify a character string to name the list you are creating. • For *method1*..., specify the actual method the authentication algorithm tries. The additional methods of authentication are used only if the previous method returns an error, not if it fails. Select one of these methods: • *enable*: Use the enable password for authentication. Before you can use this authentication method, you must define an enable password by using the **enable** *password* global configuration command. • *group radius*: Use RADIUS authentication. Before you can use this authentication method, you must configure the RADIUS server. • *line*: Use the line password for authentication. Before you can use this authentication method, you must

	Command or Action	Purpose
		define a line password. Use the **password** *password* line configuration command. • *local*: Use the local username database for authentication. You must enter username information in the database. Use the **username** *name* **password** global configuration command. • *local-case*: Use a case-sensitive local username database for authentication. You must enter username information in the database by using the **username** *password* global configuration command. • *none*: Do not use any authentication for login.
Step 5	**line** [**console** | **tty** | **vty**] *line-number* [*ending-line-number*] **Example:** `Device(config)# line 1 4`	Enters line configuration mode, and configure the lines to which you want to apply the authentication list.
Step 6	**login authentication** {**default** | *list-name*} **Example:** `Device(config-line)# login authentication default`	Applies the authentication list to a line or set of lines. • If you specify **default**, use the default list created with the **aaa authentication login** command. • For *list-name*, specify the list created with the **aaa authentication login** command.
Step 7	**end** **Example:** `Device(config-line)# end`	Exits line configuration mode and enters privileged EXEC mode.

Defining AAA Server Groups

You use the **server** group server configuration command to associate a particular server with a defined group server. You can either identify the server by its IP address or identify multiple host instances or entries by using the optional **auth-port** and **acct-port** keywords.

Follow these steps to define AAA server groups:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**radius server** *name* **Example:** `Device(config)# radius server ISE`	Specifies the name of the RADIUS server configuration for Protected Access Credential (PAC) provisioning and enters RADIUS server configuration mode. The device also supports RADIUS for IPv6.
Step 4	**address** {**ipv4** | **ipv6**} {*ip-address* | *hostname*} **auth-port** *port-number* **acct-port** *port-number* **Example:** `Device(config-radius-server)# address ipv4 10.1.1.1 auth-port 1645 acct-port 1646`	Configures the IPv4 address for the RADIUS server accounting and authentication parameters.
Step 5	**key** *string* **Example:** `Device(config-radius-server)# key cisco123`	Specifies the authentication and encryption key for all RADIUS communications between the device and the RADIUS server.
Step 6	**exit** **Example:** `Device(config-radius-server)# exit`	Exits RADIUS server configuration mode and enters global configuration mode.
Step 7	**aaa group server radius** *group_name* **Example:** `Device(config)# aaa group server radius abc`	Defines the RADIUS server group configuration and enters RADIUS server group configuration mode.
Step 8	**server name** *server* **Example:** `Device(config-sg-radius)# server name ISE`	Associates the RADIUS server to the server group.
Step 9	**end** **Example:** `Device(config-sg-radius)# end`	Exits RADIUS server group configuration mode and returns to privileged EXEC mode.

Configuring RADIUS Authorization for User Privileged Access and Network Services

Note Authorization is bypassed for authenticated users who log in through the CLI even if authorization has been configured.

Follow these steps to configure RADIUS authorization for user priviledged access and network services:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa authorization network** *authorization-list* **radius** **Example:** `Device(config)# aaa authorization network list1 radius`	Configures the device for user RADIUS authorization for all network-related service requests.
Step 4	**aaa authorization exec** *authorization-list* **radius** **Example:** `Device(config)# aaa authorization exec list1 radius`	Configures the device for user RADIUS authorization if the user has privileged EXEC access. The **exec** keyword might return user profile information (such as **autocommand** information).
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

What to do next

You can use the **aaa authorization** global configuration command with the **radius** keyword to set parameters that restrict a user's network access to privileged EXEC mode.

The **aaa authorization exec radius local** command sets these authorization parameters:

- Use RADIUS for privileged EXEC access authorization if authentication was performed by using RADIUS.

• Use the local database if authentication was not performed by using RADIUS.

Starting RADIUS Accounting

Follow these steps to start RADIUS accounting:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa accounting network** *accounting-list* **start-stop radius** **Example:** `Device(config)# aaa accounting network start-stop radius`	Enables RADIUS accounting for all network-related service requests.
Step 4	**aaa accounting exec** *accounting-list* **start-stop radius** **Example:** `Device(config)# aaa accounting exec acc-list start-stop radius`	Enables RADIUS accounting to send a start-record accounting notice at the beginning of a privileged EXEC process and a stop-record at the end.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Settings for All RADIUS Servers

Beginning in privileged EXEC mode, follow these steps to configure settings for all RADIUS servers:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**radius server** *server name* **Example:** `Device(config)# radius server rsim`	Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode.
Step 4	**key** *string* **Example:** `Device(config-radius-server)# key your_server_key`	Specifies the shared secret text string used between the switch and all RADIUS servers. **Note** The key is a text string that must match the encryption key used on the RADIUS server. Leading spaces are ignored, but spaces within and at the end of the key are used. If you use spaces in your key, do not enclose the key in quotation marks unless the quotation marks are part of the key.
Step 5	**retransmit** *retries* **Example:** `Device(config-radius-server)# retransmit 5`	Specifies the number of times the switch sends each RADIUS request to the server before giving up. The default is 3; the range 1 to 1000.
Step 6	**timeout** *seconds* **Example:** `Device(config-radius-server)# timeout 3`	Specifies the number of seconds a switch waits for a reply to a RADIUS request before resending the request. The default is 5 seconds; the range is 1 to 1000.
Step 7	**end** **Example:** `Device(config-radius-server)# end`	Exits RADIUS server configuration mode and enters privileged EXEC mode.

Configuring the Device to Use Vendor-Specific RADIUS Attributes

Follow these steps to configure vendor-specific RADIUS attributes:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**radius-server vsa send** [**accounting** | **authentication**] **Example:** `Device(config)# radius-server vsa send accounting`	Enables the device to recognize and use VSAs as defined by RADIUS IETF attribute 26. • (Optional) Use the **accounting** keyword to limit the set of recognized vendor-specific attributes to only accounting attributes. • (Optional) Use the **authentication** keyword to limit the set of recognized vendor-specific attributes to only authentication attributes. If you enter this command without keywords, both accounting and authentication vendor-specific attributes are used.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring the Device for Vendor-Proprietary RADIUS Server Communication

Follow these steps to configure vendor-proprietary RADIUS server communication:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**radius server** *server name* **Example:** `Device(config)# radius server rsim`	Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode.

	Command or Action	Purpose
Step 4	**address** { **ipv4** | **ipv6** } *ip address* **Example:** `Device(config-radius-server)# address ipv4 172.24.25.10`	(Optional) Specifies the IP address of the RADIUS server.
Step 5	**non-standard** **Example:** `Device(config-radius-server)# non-standard`	Identifies that the RADIUS server using a vendor-proprietary implementation of RADIUS.
Step 6	**key** *string* **Example:** `Device(config-radius-server)# key rad123`	Specifies the shared secret text string used between the device and the vendor-proprietary RADIUS server. The device and the RADIUS server use this text string to encrypt passwords and exchange responses.
Step 7	**end** **Example:** `Device(config-radius-server)# end`	Exits RADIUS server mode and enters privileged EXEC mode.

Configuring DSCP Marking on a RADIUS Server

Follow these steps to configure DSCP marking for authentication and accounting on a radius server:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**radius server** *server_name* **Example:** `Device(config)# radius server rsim`	Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode.
Step 4	**address** { **ipv4** | **ipv6** } *ip address* [**auth-port** *auth_port_number* **acct-port** *acct_port_number*] **Example:**	(Optional) Specifies the IP address of the RADIUS server. • **auth-port** configures the port value for radius authentication server. The default value is 1812.

	Command or Action	Purpose
	`Device(config-radius-server)# address ipv4 10.1.1.1 auth-port 1645 acct-port 1646`	• **acct-port** configures the port value for radius accounting server. The default value is 1813.
Step 5	**dscp** {**acct** *dscp_acct_value* | **auth** *dscp_auth_value*} **Example:** `Device(config-radius-server)# dscp auth 10 acct 20`	Configures DSCP marking for authentication and accounting on the radius server. • **acct** configures radius DSCP marking value for accounting. The valid range is from 1 to 63. The default value is 0. • **auth** configures radius DSCP marking value for authentication. The valid range is from 1 to 63. The default value is 0.
Step 6	**key** *string* **Example:** `Device(config-radius-server)# key rad123`	Specifies the shared secret text string used between the device and the vendor-proprietary RADIUS server. The device and the RADIUS server use this text string to encrypt passwords and exchange responses.
Step 7	**end** **Example:** `Device(config-radius-server)# end`	Exits RADIUS server mode and enters privileged EXEC mode.

Configuring the Source Interface and DSCP Marking on RADIUS Server Group

Follow these steps to configure the source interface and DSCP marking for authentication and accounting on radius server groups:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa group server radius** *group_name* **Example:** `Device(config)# aaa group server radius abc`	Defines the RADIUS server group configuration and enters RADIUS server group configuration mode.

	Command or Action	Purpose
Step 4	**server name** *name* **Example:** `Device(config-sg-radius)# server name serv1`	Associates the RADIUS server to the server group.
Step 5	{**ip** | **ipv6**} **radius source-interface** *type number* **Example:** `Device(config-sg-radius)# ipv6 radius source-interface GigabitEthernet 1/1`	Specifies an interface to use for the source address in RADIUS server.
Step 6	**dscp** {**acct** *dscp_acct_value* | **auth** *dscp_auth_value*} **Example:** `Device(config-sg-radius)# dscp auth 10 acct 20`	Configures DSCP marking for authentication and accounting on the radius server group. • **acct** configures radius DSCP marking value for accounting. The valid range is from 1 to 63. The default value is 0. • **auth** configures radius DSCP marking value for authentication. The valid range is from 1 to 63. The default value is 0.
Step 7	**end** **Example:** `Device(config-radius-server)# end`	Exits RADIUS server mode and enters privileged EXEC mode.

Configuring CoA on the Device

Follow these steps to configure CoA on a device. This procedure is required.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.

	Command or Action	Purpose
Step 4	**aaa server radius dynamic-author** **Example:** `Device(config)# aaa server radius dynamic-author`	Configures the device as an authentication, authorization, and accounting (AAA) server to facilitate interaction with an external policy server, and enters dynamic authorization local server configuration mode.
Step 5	**client** {*ip-address* | *name*} [**vrf** *vrfname*] [**server-key** *string*] **Example:** `Device(config-locsvr-da-radius)# client client1 vrf vrf1`	Specifies a RADIUS client from which a device will accept CoA and disconnect requests.
Step 6	**server-key** [**0** | **7**] *string* **Example:** `Device(config-locsvr-da-radius)# server-key your_server_key`	Configures the RADIUS key to be shared between a device and RADIUS clients.
Step 7	**port** *port-number* **Example:** `Device(config-locsvr-da-radius)# port 25`	Specifies the port on which a device listens for RADIUS requests from configured RADIUS clients.
Step 8	**auth-type** {**any** | **all** | **session-key**} **Example:** `Device(config-locsvr-da-radius)# auth-type any`	Specifies the type of authorization the device uses for RADIUS clients. The client must match all the configured attributes for authorization.
Step 9	**ignore server-key** **Example:** `Device(config-locsvr-da-radius)# ignore server-key`	(Optional) Configures the device to ignore the server-key.
Step 10	**exit** **Example:** `Device(config-locsvr-da-radius)# exit`	Exits dynamic authorization local server configuration mode and returns to global configuration mode.
Step 11	**authentication command bounce-port ignore** **Example:** `Device(config)# authentication command bounce-port ignore`	(Optional) Configures the device to ignore a CoA request to temporarily disable the port hosting a session. The purpose of temporarily disabling the port is to trigger a DHCP renegotiation from the host when a VLAN change occurs and there is no supplicant on the endpoint to detect the change.
Step 12	**authentication command disable-port ignore** **Example:** `Device(config)# authentication command disable-port ignore`	(Optional) Configures the device to ignore a nonstandard command requesting that the port hosting a session be administratively shut

	Command or Action	Purpose
		down. Shutting down the port results in termination of the session. Use standard CLI or SNMP commands to re-enable the port.
Step 13	**end** **Example:** `Device(config)# end`	Exits global configuration mode an returns to privileged EXEC mode.

Configuring RADIUS Source-Interface Under a RADIUS Server-Group

The RADIUS source-interface can be configured under a RADIUS server-group in either of the following methods:

- Configure a RADIUS source-interface under the RADIUS server-group using the **ip radius source-interface** *interface-name* command.
- Configure a VRF using the **vrf** *vrf-name* command under the RADIUS server-group, and then associate the configured VRF globally to a source-interface using the **ip radius source interface** *interface-name* **vrf** *vrf-name* command.

Priority will be given to the source-interface under the server-group configuration in case both methods are configured.

To configure RADIUS source-interface under a RADIUS server-group, perform the following:

Before you begin

You must configure a VRF routing table and associate VRF to an interface

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	{ **ip** | **ipv6** }**radius source-interface** *interface-number* **vrf** *vrf-name* **Example:** `Device(config)# ip radius source-interface GigabitEthernet1/1 vrf vrf17`	Forces RADIUS to use the IP address of a specified interface for all outgoing RADIUS packets, and enables the specification on a per-VRF basis.

	Command or Action	Purpose
		• *interface-name*: Specifies the name of the interface that RADIUS uses for all of its outgoing packets. • **vrf** *vrf-name*: Specifies the per-VRF configuration.
Step 4	**aaa group server radius** *group_name* **Example:** `Device(config-sg-radius)# aa group server radius rad-grp`	Groups different RADIUS server hosts into distinct lists and distinct methods and enters server-group configuration mode.
Step 5	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-sg-radius)# ip vrf forwarding vrf17`	(Optional) Configures a VRF for the interface.
Step 6	{ **ip** | **ipv6** } **radius source-interface** *interface-number* **Example:** `Device(config-sg-radius)# ip radius source-interface loopback0`	(Optional) Forces RADIUS to use the IP address of a specified interface for all outgoing RADIUS packets from the RADIUS group server. *interface-name*: Specifies the name of the interface that RADIUS uses for all of its outgoing packets.
Step 7	**end** **Example:** `Device(config-sg-radius)# end`	Returns to privileged EXEC mode.

Monitoring CoA Functionality

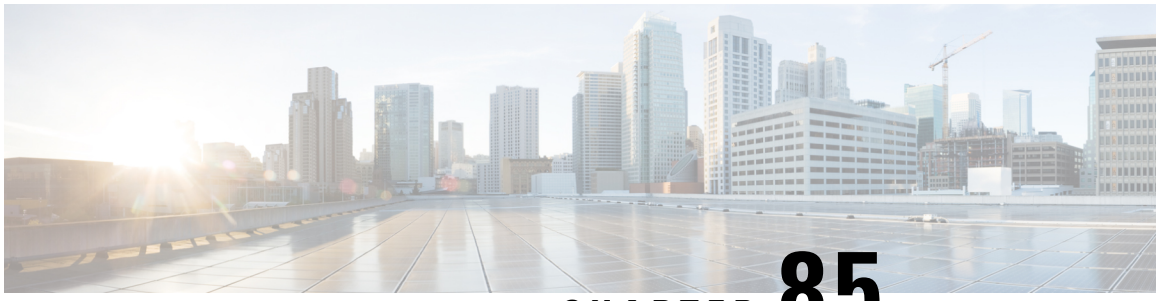
Table 98: Privileged EXEC show Commands

Command	Purpose
show aaa attributes protocol radius	Displays AAA attributes of RADIUS commands.

Table 99: Global Troubleshooting Commands

Command	Purpose
debug radius	Displays information for troubleshooting RADIUS.
debug aaa coa	Displays information for troubleshooting CoA processing.
debug aaa pod	Displays information for troubleshooting POD packets.

Command	Purpose
debug aaa subsys	Displays information for troubleshooting POD packets.
debug cmdhd [**detail** | **error** | **events**]	Displays information for troubleshooting command headers.



CHAPTER 85

Configuring RadSec

This chapter describes how to configure RadSec over Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) servers.

Restrictions for Configuring RadSec

The following restrictions apply to the RadSec feature:

- A RADIUS client uses an ephemeral port as the source port. This source port should not be used for UDP, Datagram Transport Layer Security (DTLS), and Transport Layer Security (TLS) at the same time.
- Although there is no configuration restriction, we recommend that you use the same type—either only TLS or only DTLS—for a server under an AAA server group.
- RadSec is not supported on the DTLS port range 1 to 1024.

Note DTLS ports must be configured to work with the RADIUS server.

- RadSec is not supported with high availability.
- RADIUS Change of Authorization (CoA) reception of request and transmission of response over the same authentication channel is supported with RadSec over TLS only. It is not supported over DTLS or plain RADIUS.
- The **tls watchdoginterval** command is not applicable for Packet of Disconnect (PoD) use cases.
- FQDN configuration for CoA is not supported.

Information About RadSec

RadSec provides encryption services over the RADIUS server transported over a secure tunnel. RadSec over TLS and DTLS is implemented in both client and device servers. While the client side controls RADIUS AAA, the device side controls CoA.

You can configure the following parameters:

- Individual client-specific idle timeout, client trustpoint, and server trustpoint.
- Global CoA-specific TLS or DTLS listening port and the corresponding list of source interfaces.

Note You can disable TLS or DTLS for a specific server by using the **no tls** or **no dtls** command in radius server configuration mode.

RadSec CoA request reception and CoA response transmission over the same authentication channel can be enabled by configuring the **tls watchdoginterval** command. The TLS watchdog timer must be lesser than the TLS idle timer so that the established tunnel remains active if RADIUS test authentication packets are seen before the idle timer expires. If the tunnel is torn down and **tls watchdoginterval** command is enabled, the tunnel gets re-established immediately. If **tls watchdoginterval** command is disabled, CoA requests on the same authentication channel are discarded.

How to Configure RadSec

The following sections provide information about the various tasks that comprise RadSec configuration.

Configuring RadSec over TLS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**radius server** *radius-server-name* **Example:** `Device(config)# radius server R1`	Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode.

	Command or Action	Purpose
Step 4	**tls** [**connectiontimeout** *connection-timeout-value*] [**idletimeout** *idle-timeout-value*] [[**ip** | **ipv6**] {**radius source-interface** *interface-name* |**vrf forwarding** *forwarding-table-name*}] [**match-server-identity** {**email-address** *email-address* | **hostname** *host-name* | **ip-address** *ip-address*}] [**port** *port-number*] [**retries** *number-of-connection-retries*] [**trustpoint** {**client** *trustpoint name* | **server** *trustpoint name*}] [**watchdoginterval** *interval*] **Example:** `Device(config-radius-server)# tls connectiontimeout 10` `Device(config-radius-server)# tls idletimeout 75` `Device(config-radius-server)# tls retries 15` `Device(config-radius-server)# tls ip radius source-interface GigabitEthernet 1/1` `Device(config-radius-server)# tls ipv6 vrf forwarding table-1` `Device(config-radius-server)# tls match-server-identity ip-address 10.1.1.10` `Device(config-radius-server)# tls port 10` `Device(config-radius-server)# tls trustpoint client TP-self-signed-721943660` `Device(config-radius-server)# tls trustpoint server isetp` `Device(config-radius-server)# tls watchdoginterval 10`	Configures the TLS parameters. You can configure the following parameters: • **connectiontimeout**: Configures TLS connection timeout value. The default is 5 seconds. • **idletimeout**: Configures the TLS idle timeout value. The default is 60 seconds. • **ip**: Configures IP source parameters. • **ipv6**: Configures IPv6 source parameters. • **match-server-identity**: Configures RadSec certification validation parameters. **Note** This is a mandatory configuration. • **port**: Configures the TLS port number. The default is 2083. • **retries**: Configures the number of TLS connection retries. The default is 5. • **trustpoint**: Configures the TLS trustpoint for a client and a server. If the TLS trustpoint for the client and server are the same, the trustpoint name should also be the same for both. • **watchdoginterval**: Configures the watchdog interval. This enables CoA requests to be received on the same authentication channel. It also serves as a keepalive to keep the TLS tunnel up, and re-establishes the tunnel if it is torn down. **Note** **watchdoginterval** value must be lesser than **idletimeout**, for the established tunnel to remain up.
Step 5	**end** **Example:** `Device(config-radius-server)# end`	Exits RADIUS server configuration mode and returns to privileged EXEC mode.

Configuring Dynamic Authorization for TLS CoA

Note When the **tls watchdoginterval** command is enabled, the client IP configuration under **aaa server radius dynamic-author** command is not used. Instead, the key configured under **radius server** command is used for CoA transactions.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa server radius dynamic-author** **Example:** `Device(config)# aaa server radius dynamic-author`	Enters dynamic authorization local server configuration mode and specifies the RADIUS client from which a device accepts CoA and disconnect requests. Configures the device as an AAA server to facilitate interaction with an external policy server.
Step 4	**client {ip-addr | hostname}** [**tls** [**client-tp** *client-tp-name*] [**idletimeout** *idletimeout-interval*] [**server-key** *server-key*] [**server-tp** *server-tp-name*]] **Example:** `Device(config-locsvr-da-radius)# client 10.104.49.14 tls idletimeout 100 client-tp tls_ise server-tp tls_client server-key key1`	Configures the IP address or hostname of the AAA server client. You can configure the following optional parameters: • **tls**: Enables TLS for the client. • **client-tp**: Configures the client trustpoint. • **idletimeout**: Configures the TLS idle timeout value. • **server-key**: Configures a RADIUS client server key. • **server-tp**: Configures the server trustpoint.
Step 5	**end** **Example:** `Device(config-locsvr-da-radius)# end`	Exits dynamic authorization local server configuration mode and returns to privileged EXEC mode.

Configuring RadSec over DTLS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**radius server** *radius-server-name* **Example:** `Device(config)# radius server R1`	Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode.
Step 4	**dtls** [**connectiontimeout** *connection-timeout-value*] [**idletimeout** *idle-timeout-value*] [[**ip | ipv6**] {**radius source-interface** *interface-name* |**vrf forwarding** *forwarding-table-name*}] [**match-server-identity** {**email-address** *email-address* | **hostname** *host-name* | **ip-address** *ip-address*}] [**port** *port-number*] [**retries** *number-of-connection-retries*] [**trustpoint** {**client** *trustpoint name* | **server** *trustpoint name*}] **Example:** `Device(config-radius-server)# dtls connectiontimeout 10` `Device(config-radius-server)# dtls idletimeout 75` `Device(config-radius-server)# dtls retries 15` `Device(config-radius-server)# dtls ip radius source-interface GigabitEthernet 1/1` `Device(config-radius-server)# dtls ipv6 vrf forwarding table-1` `Device(config-radius-server)# tls match-server-identity ip-address 10.1.1.10` `Device(config-radius-server)# dtls port 10` `Device(config-radius-server)# dtls trustpoint client TP-self-signed-721943660` `Device(config-radius-server)# dtls trustpoint server isetp`	Configures DTLS parameters. You can configure the following parameters: • **connectiontimeout**: Configures the DTLS connection timeout value. The default is 5 seconds. • **idletimeout**: Configures the DTLS idle timeout value. The default is 60 seconds. **Note** When the idle timeout expires, and there are no transactions after the last idle timeout, the DTLS session is closed. When the session is re-established, restart the idle timer for the session to work. If the configured idle timeout is 30 seconds, when the timeout expires, the number of RADIUS DTLS transactions are checked. If the RADIUS DTLS packets are more than 0, the transaction counter is reset and the timer is started again. • **ip**: Configures IP source parameters. • **ipv6**: Configures IPv6 source parameters. • **match-server-identity**: Configures RadSec certification validation parameters. **Note** This is a mandatory configuration.

	Command or Action	Purpose
		• **port**: Configures the DTLS port number. The default is 2083. • **retries**: Configures the number of DTLS connection retries. The default is 5. • **trustpoint**: Configures the DTLS trustpoint for the client and the server. If the DTLS trustpoint for the client and server are the same, the trustpoint name should also be the same for both.
Step 5	**end** **Example:** `Device(config-radius-server)# end`	Exits RADIUS server configuration mode and returns to privileged EXEC mode.

Configuring Dynamic Authorization for DTLS CoA

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa server radius dynamic-author** **Example:** `Device(config)# aaa server radius dynamic-author`	Enters dynamic authorization local server configuration mode and specifies a RADIUS client from which the device accepts CoA and disconnect requests. Configures the device as an AAA server to facilitate interaction with an external policy server.
Step 4	**client {ip-addr | hostname}** [**dtls** [**client-tp** *client-tp-name*] [**idletimeout** *idletimeout-interval*] [**server-key** *server-key*] [**server-tp** *server-tp-name*]] **Example:** `Device(config-locsvr-da-radius)# client 10.104.49.14 dtls idletimeout 100 client-tp tls_ise server-tp tls_client server-key key1`	Configures the IP address or hostname of the AAA server client. You can configure the following optional parameters: • **tls**: Enables TLS for the client. • **client-tp**: Configures the client trustpoint. • **idletimeout**: Configures the TLS idle timeout value.

	Command or Action	Purpose
		• **server-key**: Configures a RADIUS client server key. • **server-tp**: Configures the server trustpoint.
Step 5	**dtls** {{**ip** | **ipv6**} **radius source-interface** *interface-name* | **port** *radius-dtls-server-port-number*} **Example:** `Device(config-locsvr-da-radius)# dtls ip radius source-interface GigabitEthernet 1/1` `Device(config-locsvr-da-radius)# dtls port 100`	Configures the RADIUS CoA server. You can configure the following parameters: • {**ip** | **ipv6**} **radius source-interface** *interface-name*: Specifies the interface for the source address in the RADIUS CoA server. • **port** *radius-dtls-server-port-number*: Specifies the port on which the local DTLS RADIUS server listens.
Step 6	**end** **Example:** `Device(config-locsvr-da-radius)# end`	Exits dynamic authorization local server configuration mode and returns to privileged EXEC mode.

Monitoring RadSec

Use the following commands to monitor TLS and DTLS server statistics.

Table 100: Monitoring TLS and DTLS Server Statistics

Command	Purpose
show aaa servers	Displays information related to TLS and DTLS servers.
clear aaa counters servers radius {*server id* | **all**}	Clears the RADIUS TLS-specific or DTLS-specific statistics.
debug radius radsec	Enables RADIUS RadSec debugs.

Configuration Examples for RadSec

The following examples help you understand the RadSec configuration better.

Example: Configuring RadSec over TLS

The following example shows how to configure RadSec over TLS:

```
Device> enable
Device# configure terminal
Device(config)# radius server R1
Device(config-radius-server)# tls connectiontimeout 10
Device(config-radius-server)# tls idletimeout 75
Device(config-radius-server)# tls retries 15
Device(config-radius-server)# tls ip radius source-interface GigabitEthernet 1/1
Device(config-radius-server)# tls ip vrf forwarding table-1
Device(config-radius-server)# tls port 10
Device(config-radius-server)# tls trustpoint client TP-self-signed-721943660
Device(config-radius-server)# tls trustpoint server isetp
Device(config-radius-server)# tls watchdoginterval 10
Device(config-radius-server)# end
```

Example: Configuring Dynamic Authorization for TLS CoA

The following example shows how to configure dynamic authorization for TLS CoA:

```
Device> enable
Device# configure terminal
Device(config)# aaa server radius dynamic-author
Device(config-locsvr-da-radius)# client 10.104.49.14 tls idletimeout 100
client-tp tls_ise server-tp tls_client
Device(config-locsvr-da-radius)# end
```

Example: Configuring RadSec over DTLS

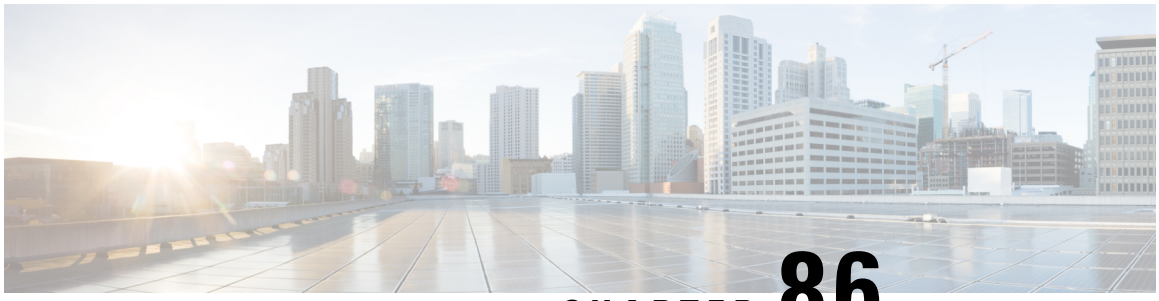
The following example shows how to configure RadSec over DTLS:

```
Device> enable
Device# configure terminal
Device(config)# radius server R1
Device(config-radius-server)# dtls connectiontimeout 10
Device(config-radius-server)# dtls idletimeout 75
Device(config-radius-server)# dtls retries 15
Device(config-radius-server)# dtls ip radius source-interface GigabitEthernet 1/1
Device(config-radius-server)# dtls ip vrf forwarding table-1
Device(config-radius-server)# dtls port 10
Device(config-radius-server)# dtls trustpoint client TP-self-signed-721943660
Device(config-radius-server)# dtls trustpoint server isetp
Device(config-radius-server)# end
```

Example: Configuring Dynamic Authorization for DTLS CoA

The following example shows how to configure dynamic authorization for DTLS CoA:

```
Device> enable
Device# configure terminal
Device(config)# aaa server radius dynamic-author
Device(config-locsvr-da-radius)# client 10.104.49.14 dtls idletimeout 100
client-tp dtls_ise server-tp dtls_client
Device(config-locsvr-da-radius)# dtls ip radius source-interface GigabitEthernet 1/1
Device(config-locsvr-da-radius)# dtls port 100
Device(config-locsvr-da-radius)# end
```



CHAPTER 86

Configuring RADIUS Server Load Balancing

The RADIUS Server Load Balancing feature distributes authentication, authorization, and accounting (AAA) authentication and accounting transactions across RADIUS servers in a server group. These servers can share the AAA transaction load and thereby respond faster to incoming requests.

This module describes the RADIUS Server Load Balancing feature.

Prerequisites for RADIUS Server Load Balancing

- Authentication, authorization, and accounting (AAA) must be configured on the RADIUS server.
- AAA RADIUS server groups must be configured.
- RADIUS must be configured for functions such as authentication, accounting, or static route download.

Restrictions for RADIUS Server Load Balancing

- Incoming RADIUS requests, such as Packet of Disconnect (POD) requests, are not supported.
- Load balancing is not supported on proxy RADIUS servers and for private server groups.

Information About RADIUS Server Load Balancing

RADIUS Server Load Balancing Overview

Load balancing distributes batches of transactions to RADIUS servers within a server group. Load balancing assigns each batch of transactions to the server with the lowest number of outstanding transactions in its queue. The process of assigning a batch of transactions is as follows:

1. The first transaction is received for a new batch.
2. All server transaction queues are checked.
3. The server with the lowest number of outstanding transactions is identified.
4. The identified server is assigned the next batch of transactions.

The batch size is a user-configured parameter. Changes in the batch size may impact CPU load and network throughput. As batch size increases, CPU load decreases and network throughput increases. However, if a large batch size is used, all available server resources may not be fully utilized. As batch size decreases, CPU load increases and network throughput decreases.

Note There is no set number for large or small batch sizes. A batch with more than 50 transactions is considered large and a batch with fewer than 25 transactions is considered small.

Note If a server group contains ten or more servers, we recommend that you set a high batch size to reduce CPU load.

Transaction Load Balancing Across RADIUS Server Groups

You can configure load balancing either per-named RADIUS server group or for the global RADIUS server group. The load balancing server group must be referred to as “radius” in the authentication, authorization, and accounting (AAA) method lists. All public servers that are part of the RADIUS server group are then load balanced.

You can configure authentication and accounting to use the same RADIUS server or different servers. In some cases, the same server can be used for preauthentication, authentication, or accounting transactions for a session. The preferred server, which is an internal setting and is set as the default, informs AAA to use the same server for the start and stop record for a session regardless of the server cost. When using the preferred server setting, ensure that the server that is used for the initial transaction (for example, authentication), the preferred server, is part of any other server group that is used for a subsequent transaction (for example, accounting).

The preferred server is not used if one of the following criteria is true:

- The **load-balance method least-outstanding ignore-preferred-server** command is used.
- The preferred server is dead.
- The preferred server is in quarantine.
- The want server flag has been set, overriding the preferred server setting.

The want server flag, an internal setting, is used when the same server must be used for all stages of a multistage transaction regardless of the server cost. If the want server is not available, the transaction fails.

You can use the **load-balance method least-outstanding ignore-preferred-server** command if you have either of the following configurations:

- Dedicated authentication server and a separate dedicated accounting server

- Network where you can track all call record statistics and call record details, including start and stop records and records that are stored on separate servers

If you have a configuration where authentication servers are a superset of accounting servers, the preferred server is not used.

RADIUS Server Status and Automated Testing

The RADIUS Server Load Balancing feature considers the server status when assigning batches. Transaction batches are sent only to live servers. We recommend that you test the status of all RADIUS load-balanced servers, including low usage servers (for example, backup servers).

Transactions are not sent to a server that is marked dead. A server is marked dead until its timer expires, at which time it moves to quarantine state. A server is in quarantine until it is verified alive by the RADIUS automated tester functionality.

To determine if a server is alive and available to process transactions, the RADIUS automated tester sends a request periodically to the server for a test user ID. If the server returns an Access-Reject message, the server is alive; otherwise the server is either dead or quarantined.

A transaction sent to an unresponsive server is failed over to the next available server before the unresponsive server is marked dead. We recommend that you use the retry reorder mode for failed transactions.

When using the RADIUS automated tester, verify that the authentication, authorization, and accounting (AAA) servers are responding to the test packets that are sent by the network access server (NAS). If the servers are not configured correctly, packets may be dropped and the server erroneously marked dead.

Caution We recommend that you use a test user that is not defined on the RADIUS server for the RADIUS server automated testing to protect against security issues that may arise if the test user is not correctly configured.

Note Use the **test aaa group** command to check load-balancing transactions.

The **automate-tester username** *name* **probe-on** command is used to verify the status of a server by sending RADIUS packets. After this command is configured, a five-second dead timer is started and a RADIUS packet is sent to the external RADIUS server after five seconds. The server state is updated if there is a response from the external RADIUS server. If there is no response, the packets are sent out according to the timeout interval that is configured using the **radius-server timeout** command. This will continue for 180 seconds, and if there is still no response, a new dead timer is started based on the configured **radius-server deadtime** command.

VRF-Aware RADIUS Automated Testing

The RADIUS automated tester function works at a server-level configuration. There is no group associated with the function. A VRF is a group level configuration. All the information related to the VRF and the source-interface configurations is maintained in a group structure. If information regarding the VRF and the source-interface configurations is available in the global source-interface, automated tester can access it. If the information is not available at the global source-interface or the default VRF, automated tester marks the server as a dead server.

You can configure automated tester to be VRF aware. Use the **vrf** keyword with the **automate-tester** command to enable automate-tester for a non-default VRF.

Note For VRF aware automate-tester to work, you have to configure **global config ipv4/ipv6 source interface** *interface-name* **vrf** *vrf-name* command.

How to Configure RADIUS Server Load Balancing

Enabling Load Balancing for a Named RADIUS Server Group

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa group server radius** *group-name* **Example:** `Device(config)# aaa group server radius rad-sg`	Enters server group configuration mode.
Step 4	**server ip-address [auth-port port-number] [acct-port port-number]** **Example:** `Device(config-sg-radius)# server 192.0.2.238 auth-port 2095 acct-port 2096`	Configures the IP address of the RADIUS server for the group server.
Step 5	**load-balance method least-outstanding** [**batch-size** *number*] [**ignore-preferred-server**] **Example:** `Device(config-sg-radius)# load-balance method least-outstanding batch-size 30`	Enables the least-outstanding load balancing for a named server group.
Step 6	**end** **Example:** `Device(config-sg-radius)# end`	Exits server group configuration mode and returns to privileged EXEC mode.

Troubleshooting RADIUS Server Load Balancing

After configuring the RADIUS Server Load Balancing feature, you can monitor the idle timer, dead timer, and load balancing server selection or verify the server status by using a manual test command.

Procedure

Step 1 Use the **debug aaa test** command to determine when an idle timer or dead timer has expired, when test packets are sent, the status of the server, or to verify the server state.

The idle timer is used to check the server status and is updated with or without any incoming requests. Monitoring the idle timer helps to determine if there are nonresponsive servers and to keep the RADIUS server status updated to efficiently utilize available resources. For instance, an updated idle timer would help ensure that incoming requests are sent to servers that are alive.

The dead timer is used either to determine that a server is dead or to update a dead server's status appropriately.

Monitoring server selection helps to determine how often the server selection changes. Server selection is effective in analyzing if there are any bottlenecks, a large number of queued requests, or if only specific servers are processing incoming requests.

The following sample output from the **debug aaa test** command shows when the idle timer expired:

Example:

```
Device# debug aaa test

Jul 16 00:07:01: AAA/SG/TEST: Server (192.0.2.245:1700,1701) quarantined.
Jul 16 00:07:01: AAA/SG/TEST: Sending test request(s) to server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Sending 1 Access-Requests, 1 Accounting-Requests in current
batch.
Jul 16 00:07:01: AAA/SG/TEST(Req#: 1): Sending test AAA Access-Request.
Jul 16 00:07:01: AAA/SG/TEST(Req#: 1): Sending test AAA Accounting-Request.
Jul 16 00:07:01: AAA/SG/TEST: Obtained Test response from server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Obtained Test response from server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Necessary responses received from server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Server (192.0.2.245:1700,1701) marked ALIVE. Idle timer set
for 60 sec(s).
Jul 16 00:07:01: AAA/SG/TEST: Server (192.0.2.245:1700,1701) removed from quarantine.
```

Step 2 Use the **debug aaa sg-server selection** command to determine the server that is selected for load balancing.

The following sample output from the **debug aaa sg-server selection** command shows five access requests being sent to a server group with a batch size of three:

Example:

```
Device# debug aaa sg-server selection

Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [3] transactions remaining in batch. Reusing server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [2] transactions remaining in batch. Reusing server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [1] transactions remaining in batch. Reusing server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: No more transactions in batch. Obtaining a new server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining a new least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Server[0] load: 3
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Server[1] load: 0
```

```
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Server[2] load: 0
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Selected Server[1] with load 0
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [3] transactions remaining in batch.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [2] transactions remaining in batch. Reusing server.
```

Step 3 Use the **test aaa group** command to manually verify the RADIUS load-balanced server status.

The following sample output shows the response from a load-balanced RADIUS server that is alive when the username "test" does not match a user profile. The server is verified alive when it issues an Access-Reject response to an authentication, authorization, and accounting (AAA) packet generated using the **test aaa group** command.

Example:

```
Device# test aaa group SG1 test lab new-code

00:06:07: RADIUS/ENCODE(00000000):Orig. component type = INVALID
00:06:07: RADIUS/ENCODE(00000000): dropping service type, "radius-server attribute 6
on-for-login-auth" is off
00:06:07: RADIUS(00000000): Config NAS IP: 192.0.2.4
00:06:07: RADIUS(00000000): sending
00:06:07: RADIUS/ENCODE: Best Local IP-Address 192.0.2.141 for Radius-Server 192.0.2.176
00:06:07: RADIUS(00000000): Send Access-Request to 192.0.2.176:1645 id 1645/1, len 50
00:06:07: RADIUS:  authenticator CA DB F4 9B 7B 66 C8 A9 - D1 99 4E 8E A4 46 99 B4
00:06:07: RADIUS:  User-Password       [2]   18  *
00:06:07: RADIUS:  User-Name           [1]   6   "test"
00:06:07: RADIUS:  NAS-IP-Address      [4]   6   192.0.2.141
00:06:07: RADIUS: Received from id 1645/1 192.0.2.176:1645, Access-Reject, len 44
00:06:07: RADIUS:  authenticator 2F 69 84 3E F0 4E F1 62 - AB B8 75 5B 38 82 49 C3
00:06:07: RADIUS:  Reply-Message       [18]  24
00:06:07: RADIUS:   41 75 74 68 65 6E 74 69 63 61 74 69 6F 6E 20 66  [Authentication f]
00:06:07: RADIUS:   61 69 6C 75 72 65                               [failure]
00:06:07: RADIUS(00000000): Received from id 1645/1
00:06:07: RADIUS/DECODE: Reply-Message fragments, 22, total 22 bytes
```

Enabling VRF Aware RADIUS Automated Testing

To enable RADIUS automated testing for a non-default VRF, perform the following procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**radius server** *name* **Example:** `Device(config)# radius server myserver`	Specifies the name of the RADIUS server configuration and enters RADIUS server configuration mode.
Step 4	**address** {**ipv4** | **ipv6**} {*ip-address* | *host-name*} **auth-port** *port-numer* **acct-port** *port-number* **Example:** `Device(config-radius-server)# address ipv4 192.0.2.1 auth-port 1812 acct-port 1813`	Configures the IPv4 address for the RADIUS server accounting and authentication parameters.
Step 5	**automate-tester username** *user* [**ignore-auth-port**] [**ignore-acct-port**] [**idle-time** *minutes*] **vrf** *vrf-name* or **automate-tester username** *user* **probe-on vrf** *vrf-name* **Example:** `Device(config-radius-server)# automate-tester username user1 idle-time 2 vrf VRF1` or `Device(config-radius-server)# automate-tester username user1 probe-on vrf VRF1`	Enables RADIUS automated testing for a non-default VRF.
Step 6	**end** **Example:** `Device(config-radius-server)# end`	Exits RADIUS server configuration mode and returns to privileged EXEC mode.

Configuration Examples for RADIUS Server Load Balancing

Example: Enabling Load Balancing for a Named RADIUS Server Group

The following examples show load balancing enabled for a named RADIUS server group. These examples are shown in three parts: the current configuration of the RADIUS command output, debug output, and authentication, authorization, and accounting (AAA) server status information.

The following sample output shows the relevant RADIUS configuration:

```
Device# show running-config
.
.
.
aaa group server radius server-group1
 server 192.0.2.238 auth-port 2095 acct-port 2096
 server 192.0.2.238 auth-port 2015 acct-port 2016
```

```
 load-balance method least-outstanding batch-size 5
!
aaa authentication dot1x default group server-group1
aaa accounting network default start-stop group server-group1
.
.
.
Device(config-sg-radius)# load-balance method least-outstanding batch-size 30
```

The lines in the current configuration of the preceding RADIUS command output are defined as follows:

- The **aaa group server radius** command shows the configuration of a server group with two member servers.
- The **load-balance** command enables load balancing for global RADIUS server groups with the batch size specified.
- The **aaa accounting** command enables sending of all accounting requests to the AAA server when the client is authenticated and then disconnected using the **start-stop** keyword.

The show debug sample output below shows the selection of the preferred server and the processing of requests for the preceding configuration:

```
Device# show debug

*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002C):No preferred server available.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:No more transactions in batch. Obtaining a new
server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining a new least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Server[0] load:0
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Server[1] load:0
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Selected Server[0] with load 0
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:[5] transactions remaining in batch.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002C):Server (192.0.2.238:2095,2096) now being
 used as preferred server
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002D):No preferred server available.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:[4] transactions remaining in batch. Reusing
server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002D):Server (192.0.2.238:2095,2096) now being
 used as preferred server
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002E):No preferred server available.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:[3] transactions remaining in batch. Reusing
server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002E):Server (192.0.2.238:2095,2096) now being
 used as preferred server
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002F):No preferred server available.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:[2] transactions remaining in batch. Reusing
server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002F):Server (192.0.2.238:2095,2096) now being
 used as preferred server
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(00000030):No preferred server available.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:[1] transactions remaining in batch. Reusing
server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(00000030):Server (192.0.2.238:2095,2096) now being
 used as preferred server
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT(00000031):No preferred server available.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:No more transactions in batch. Obtaining a new
```

```
server.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Obtaining a new least loaded server.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Server[1] load:0
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Server[0] load:5
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Selected Server[1] with load 0
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:[5] transactions remaining in batch.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT(00000031):Server (192.0.2.238:2015,2016) now being
 used as preferred server
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT(00000032):No preferred server available.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:[4] transactions remaining in batch. Reusing
server.
.
.
.
```

The following sample output from the **show aaa servers** command shows the AAA server status for the named RADIUS server group configuration:

The sample output shows the status of two RADIUS servers. Both servers are alive, and no requests have been processed since the counters were cleared 0 minutes ago.

```
Device# show aaa servers

RADIUS: id 3, priority 1, host 9:76:239::219, auth-port 1812, acct-port 1813, hostname r6
State: current UP, duration 223000s, previous duration 301s
Dead: total time 682s, count 2
Platform State from SMD: current UP, duration 222972s, previous duration 258s
SMD Platform Dead: total time 702s, count 3
Platform State from WNCD (1) : current UP
Platform State from WNCD (2) : current UP
Platform State from WNCD (3) : current UP
Platform State from WNCD (4) : current UP
Platform State from WNCD (5) : current UP
Platform State from WNCD (6) : current UP
Platform State from WNCD (7) : current UP
Platform State from WNCD (8) : current UP, duration 2451264s, previous duration 258s
Platform Dead: total time 703s, count 3
Quarantined: No
Authen: request 68, timeouts 68, failover 0, retransmission 53


Sates defination:
State: current UP. ===> this is IOSD state
Platform State from SMD: current UP. ====> This is wired BINOS i,e SMD
Platform State from WNCD (1) : current UP ===> This is wireless BINOS i.e WNCD instance 1
Platform State from WNCD (2) : current UP. ===> This is wireless BINOS i.e WNCD instance 2
Platform State from WNCD (3) : current UP
Platform State from WNCD (4) : current UP
Platform State from WNCD (5) : current UP
Platform State from WNCD (6) : current UP
Platform State from WNCD (7) : current UP
Platform State from WNCD (8) : current UP. ===> This is wireless BINOS i.e WNCD instance 8
```

Example: Monitoring Idle Timer

The following example shows idle timer and related server state for load balancing enabled for a named RADIUS server group. The current configuration of the RADIUS command output and debug command output are also displayed.

The following sample output shows the relevant RADIUS configuration:

```
Device(config)# do show run aaa

aaa group server radius server-group1
radius server server1
address ipv4 192.0.2.1 auth-port 1812 acct-port 1813
automate-tester username user1 idle-time 2 vrf VRF1
radius-server load-balance method least-outstanding batch-size 5
```

The lines in the current configuration of the preceding RADIUS command output are defined as follows:

- The **aaa group server radius** command shows the configuration of a server group.
- The **radius server** and **address** command defines the RADIUS server name and IP address of the RADIUS server with authorization and accounting ports specified.
- The **radius-server load-balance** command enables load balancing for the RADIUS server with the batch size specified.

The **show debug** sample output below shows test requests being sent to servers. The response to the test request sent to the server is received, the server is removed from quarantine as appropriate, the server is marked alive, and then the idle timer is reset.

```
Device# show debug

*Feb 28 13:52:20.835:AAA/SG/TEST:Server (192.0.2.238:2015,2016) quarantined.
*Feb 28 13:52:20.835:AAA/SG/TEST:Sending test request(s) to server (192.0.2.238:2015,2016)
*Feb 28 13:52:20.835:AAA/SG/TEST:Sending 1 Access-Requests, 1 Accounting-Requests in current
 batch.
*Feb 28 13:52:20.835:AAA/SG/TEST(Req#:1):Sending test AAA Access-Request.
*Feb 28 13:52:20.835:AAA/SG/TEST(Req#:1):Sending test AAA Accounting-Request.
*Feb 28 13:52:21.087:AAA/SG/TEST:Obtained Test response from server (192.0.2.238:2015,2016)
*Feb 28 13:52:22.651:AAA/SG/TEST:Obtained Test response from server (192.0.2.238:2015,2016)
*Feb 28 13:52:22.651:AAA/SG/TEST:Necessary responses received from server
(192.0.2.238:2015,2016)
*Feb 28 13:52:22.651:AAA/SG/TEST:Server (192.0.2.238:2015,2016) marked ALIVE. Idle timer
set for 60 secs(s).
*Feb 28 13:52:22.651:AAA/SG/TEST:Server (192.0.2.238:2015,2016) removed from quarantine.
.
.
.
```

Example: Configuring the Preferred Server with the Same Authentication and Authorization Server

The following example shows an authentication server group and an authorization server group that use the same servers 209.165.200.225 and 209.165.200.226. Both server groups have the preferred server flag enabled.

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius authentication-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# exit
Device(config)# aaa group server radius accounting-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# end
```

When a preferred server is selected for a session, all transactions for that session will continue to use the original preferred server. The servers 209.165.200.225 and 209.165.200.226 are load balanced based on sessions rather than transactions.

Example: Configuring the Preferred Server with Different Authentication and Authorization Servers

The following example shows an authentication server group that uses servers 209.165.200.225 and 209.165.200.226 and an authorization server group that uses servers 209.165.201.1 and 209.165.201.2. Both server groups have the preferred server flag enabled.

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius authentication-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# exit
Device(config)# aaa group server radius accounting-group
Device(config-sg-radius)# server 209.165.201.1 key radkey3
Device(config-sg-radius)# server 209.165.201.2 key radkey4
Device(config-sg-radius)# end
```

The authentication server group and the accounting server group do not share any common servers. A preferred server is never found for accounting transactions; therefore, authentication and accounting servers are load-balanced based on transactions. Start and stop records are sent to the same server for a session.

Example: Configuring the Preferred Server with Overlapping Authentication and Authorization Servers

The following example shows an authentication server group that uses servers 209.165.200.225, 209.165.200.226, and 209.165.201.1 and an accounting server group that uses servers 209.165.201.1 and 209.165.201.2. Both server groups have the preferred server flag enabled.

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius authentication-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# server 209.165.201.1 key radkey3
Device(config-sg-radius)# exit
Device(config)# aaa group server radius accounting-group
Device(config-sg-radius)# server 209.165.201.1 key radkey3
Device(config-sg-radius)# server 209.165.201.2 key radkey4
Device(config-sg-radius)# end
```

If all servers have equal transaction processing capability, one-third of all authentication transactions are directed toward the server 209.165.201.1. Therefore, one-third of all accounting transactions are also directed toward the server 209.165.201.1. The remaining two-third of accounting transactions are load balanced equally between servers 209.165.201.1 and 209.165.201.2. The server 209.165.201.1 receives fewer authentication transactions because the server 209.165.201.1 has outstanding accounting transactions.

Example: Configuring the Preferred Server with Authentication Servers As a Subset of Authorization Servers

The following example shows an authentication server group that uses servers 209.165.200.225 and 209.165.200.226 and an authorization server group that uses servers 209.165.200.225, 209.165.200.226, and 209.165.201.1. Both server groups have the preferred server flag enabled.

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius authentication-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# exit
Device(config)# aaa group server radius accounting-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# server 209.165.201.1 key radkey3
Device(config-sg-radius)# end
```

One-half of all authentication transactions are sent to the server 209.165.200.225 and the other half to the server 209.165.200.226. Servers 209.165.200.225 and 209.165.200.226 are preferred servers for authentication and accounting transaction. Therefore, there is an equal distribution of authentication and accounting transactions across servers 209.165.200.225 and 209.165.200.226. The server 209.165.201.1 is relatively unused.

Example: Configuring the Preferred Server with Authentication Servers As a Superset of Authorization Servers

The following example shows an authentication server group that uses servers 209.165.200.225, 209.165.200.226, and 209.165.201.1 and an authorization server group that uses servers 209.165.200.225 and 209.165.200.226. Both server groups have the preferred server flag enabled.

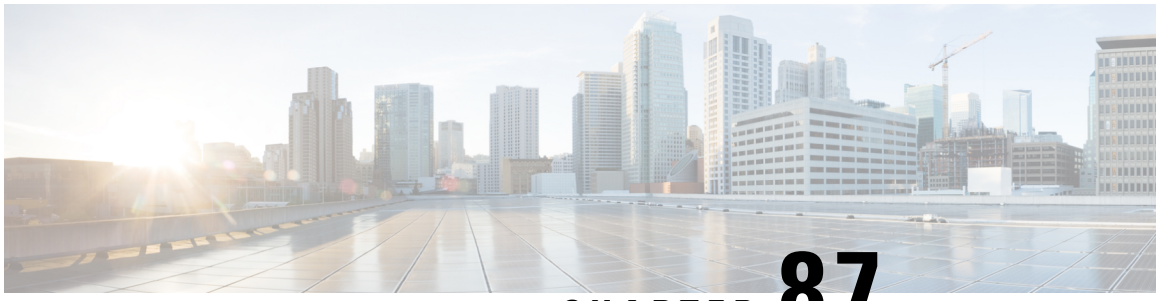
```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius authentication-groupp
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# server 209.165.201.1 key radkey3
Device(config-sg-radius)# exit
Device(config)# aaa group server radius accounting-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# end
```

Initially, one-third of authentication transactions are assigned to each server in the authorization server group. As accounting transactions are generated for more sessions, accounting transactions are sent to servers 209.165.200.225 and 209.165.200.226 because the preferred server flag is on. As servers 209.165.200.225 and 209.165.200.226 begin to process more transactions, authentication transactions will start to be sent to server 209.165.201.1. Transaction requests authenticated by server 209.165.201.1 do not have any preferred server setting and are split between servers 209.165.200.225 and 209.165.200.226, which negates the use of the preferred server flag. This configuration should be used cautiously.

Example: Enabling VRF Aware RADIUS Automated Testing

The following examples show how to enable automated testing for a non-default VRF on the RADIUS server:

```
Device(config)# radius server myserver
Device(config-radius-server)# address ipv4 192.0.2.1 auth-port 1812 acct-port 1813
Device(config-radius-server)# automate-tester username user1 idle-time 2 vrf VRF1
Device(config-radius-server)# end

Device(config)# radius server myserver
Device(config-radius-server)# address ipv4 192.0.2.1 auth-port 1812 acct-port 1813
Device(config-radius-server)# automate-tester username user1 probe-on vrf VRF1
Device(config-radius-server)# end
```



CHAPTER 87

Configuring VLAN RADIUS Attributes

The VLAN RADIUS Attributes in Access Requests feature enhances the security for access switches with the use of VLAN RADIUS attributes (VLAN name and ID) in the access requests and with an extended VLAN name length of 128 characters.

Restrictions for VLAN RADIUS Attributes in Access Requests

- Dynamic VLAN assignment to critical authentication (inaccessible authentication bypass or AAA fail policy) VLAN is not supported.
- If the RADIUS server becomes unavailable during an 802.1x authentication exchange, the current exchange times out, and the switch uses critical access control lists (ACLs) during the next authentication attempt.

Information About VLAN RADIUS Attributes in Access Requests

VLAN RADIUS Attributes

The VLAN RADIUS Attributes in Access Requests feature enhances the security for access switches with the use of VLAN RADIUS attributes (VLAN name and ID) in the access requests and with an extended VLAN name length of 128 characters.

Authentication prevents unauthorized devices (clients) from gaining access to the network by using different methods to define how users are authorized and authenticated for network access. To enhance security, you can limit network access for certain users by using VLAN assignment. Information available in the access-request packets sent to the authentication server (AAA or RADIUS server) validates the identity of the user and defines if a user can be allowed to access the network.

The VLAN RADIUS Attributes in Access Requests feature supports authentication using IEEE 802.1X, MAC authentication bypass (MAB), and web-based authentication (webauth). The default order for authentication

methods is 802.1X, and then MAB, then web-based authentication. If required, you can change the order or disable any of these methods.

- If MAC authentication bypass is enabled, the network device relays the client's MAC address to the AAA server for authorization. If the client's MAC address is valid, the authorization succeeds and the network device grants the client access to the network.
- If web-based authentication is enabled, the network device sends an HTTP login page to the client. The network device relays the client's username and password to the AAA server for authorization. If the login succeeds, the network device grants the client access to the network.

While performing authentications, the VLAN RADIUS attributes (name and ID of the VLAN) assigned to the hosting port is included in the RADIUS access requests and accounting requests. The VLAN RADIUS Attributes in Access Requests feature supports VLAN names accommodating 128-character strings.

With the use of VLAN RADIUS attributes in authentication requests, clients are authorized based on existing VLAN segmented networks. The existing VLAN provisioning is used as an indication of the location.

Based on RFC 2868 (RADIUS Attributes for Tunnel Protocol Support), support is provided for standard RADIUS attributes that exist for specifying the tunnel-type, medium and identifier.

- Tunnel-Type (IEFT #64) = VLAN
- Tunnel-Medium-Type (IEFT #65) = 802 (6)
- Tunnel-Private-Group-ID (IEFT #81) = [tag, string]

Note The Tunnel-Private-Group-ID includes the VLAN ID or name, and accommodates a string length of up to 253 characters.

How to Configure VLAN RADIUS Attributes in Access Requests

Configuring VLAN RADIUS Attributes in Access Requests

To create an attribute filter-list and to bind an attribute filter-list with authentication and accounting requests, perform the following task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**access-session attributes filter-list list** *list-name* **Example:** `Device(config)# access-session attributes filter-list list mylist`	Adds access-session protocol data to accounting and authentication records and enters common filter list configuration mode. The filter-list keyword configures a sensor protocol filter list to accounting and authentication records.
Step 4	**configure terminal** **Example:** `Device(config-com-filter-list)# vlan-id`	Includes the VLAN ID for the attribute.
Step 5	**exit** **Example:** `Device(config-com-filter-list)# exit`	Exits common filter list configuration mode and returns to global configuration mode.
Step 6	**access-session accounting attributes filter-spec include list** *list-name* **Example:** `Device(config)# access-session authentication attributes filter-spec include list mylist`	Configures a sensor protocol filter specification, and binds an attribute filter list with authentication records.
Step 7	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Verifying VLAN RADIUS Attributes in Access Requests

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**set platform software trace** *process slot module trace-level* **Example:** `Device# set platform software trace smd switch active R0 radius debug`	Sets the trace level to debug VLAN RADIUS.
Step 3	**end** **Example:** `Device# end`	Exits privileged exec mode.

Configuration Examples for VLAN RADIUS Attributes in Access Requests

Example: Configuring VLAN RADIUS Attributes in Access Requests

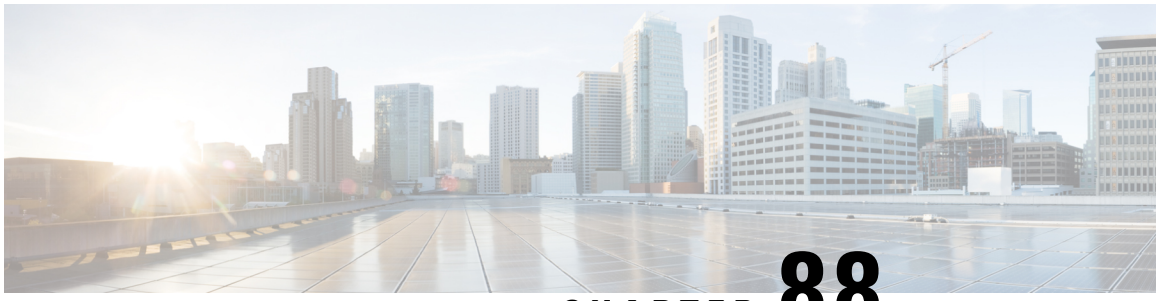
```
Device> enable
Device# configure terminal
Device(config)# access-session attributes filter-list list test-vlan-extension
Device(config-com-filter-list)# vlan-id
Device(config-com-filter-list)# end
Device(config)# access-session accounting attributes filter-spec include list mylist
Device(config)# access-session authentication attributes filter-spec include list mylist
Device(config)# end
```

Example: Verifying VLAN RADIUS Attributes in Access Requests

The following is sample output from the **set platform software trace** command. The output provides debugging information used to verify VLAN RADIUS attributes in access requests.

```
Device# set platform software trace smd switch active R0 radius debug

2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS: Send Access-Request to 10.64.69.253:1812 id 1812/38, len 296
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  authenticator 7b d2 a9 25 35 ba 1e 78 - 09 bb a8 83 02 11 b3 9d
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  User-Name           [1]      6  "hack"
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (debug):
RADIUS:  Service-Type        [6]      6  Framed                    [2]
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (debug):
RADIUS:  Vendor, Cisco       [26]    27
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:   Cisco AVpair       [1]     21  "service-type=Framed"
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  Framed-MTU          [12]     6  1468
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  EAP-Message         [79]    11  ...
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (debug):
RADIUS:  EAP-Message         [79]    11RADIUS:   02 01 00 09 01 68 61 63 6b             [
 hack]
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  Message-Authenticator[80]    18  ...
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (debug):
RADIUS:  Message-Authenticator[80]    18RADIUS:   ea c3 dd 57 ef c2 1d 4e 46 ca ea 24 ff
1d 01 aa             [ WNF$]
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  EAP-Key-Name        [102]    2  *
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (debug):
RADIUS:  EAP-Key-Name        [102]    2  *
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (debug):
RADIUS:  Vendor, Cisco       [26]    49
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:   Cisco AVpair       [1]     43  "audit-session-id=09170C33000000145B8DFD4A"
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (debug):
RADIUS:  Vendor, Cisco       [26]    20
```

```
2019/03/08 04:27:05.948 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:   Cisco AVpair       [1]     14  "method=dot1x"
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (debug):
RADIUS:  Vendor, Cisco       [26]    31
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:   Cisco AVpair       [1]     25  "client-iif-id=306305245"
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info): 01:
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  Tunnel-Private-Group-Id[81]     6  "123"
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info): 01:
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  Tunnel-Type         [64]     6  VLAN                  [13]
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info): 01:
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  Tunnel-Medium-Type  [65]     6  ALL_802               [6]
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info): 02:
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  Tunnel-Private-Group-Id[81]    11  "VLAN0123"
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info): 02:
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  Tunnel-Type         [64]     6  VLAN                  [13]
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info): 02:
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  Tunnel-Medium-Type  [65]     6  ALL_802               [6]
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  NAS-IP-Address      [4]      6  9.23.12.51
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  NAS-Port-Id         [87]    22  "GigabitEthernet1/1"
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  NAS-Port-Type       [61]     6  Ethernet                [15]
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  NAS-Port            [5]      6  50106
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (info):
RADIUS:  Calling-Station-Id  [31]    19  "20-37-06-CF-B7-18"
2019/03/08 04:27:05.949 {smd_R0-0}{1}: [radius] [17946]: UUID: 0, ra: 0, TID: 0 (debug):
RADIUS(00000000): Sending a IPv4 Radius Packet
```



CHAPTER 88

Configuring MACsec Encryption

Prerequisites for MACsec Encryption

Prerequisites for MACsec Encryption

This section list the prerequisites for MACsec encryption:

- Enable the **ssci-based-on-sci** command while configuring MACsec encryption on the device to allow interoperability with non-Cisco and non-IOS XE devices.
- Ensure that 802.1x authentication and AAA are configured on your device.
- You must configure the **flowcontrol receive desired** command on all MACsec-enabled ports to enable flowcontrol explicitly.

Prerequisites for Certificate-Based MACsec

This section list the prerequisites for Certificate-Based MACsec:

- Ensure that you have a Certificate Authority (CA) server configured for your network.
- Generate a CA certificate.
- Ensure that you have configured Cisco Identity Services Engine (ISE) Release 2.0.
- Ensure that both the participating devices, the CA server, and Cisco Identity Services Engine (ISE) are synchronized using Network Time Protocol (NTP). If time is not synchronized on all your devices, certificates will not be validated.

Restrictions for MACsec Encryption

- For IE3505 switches, MACsec is not supported with the FPGA features like HSR/PRP, DLR.
- The Network Essentials package supports MACsec with 128-bit encryption, while the Network Advantage package supports MACsec with 256-bit encryption.
- MACsec with MACsec Key Agreement (MKA) is supported only on point-to-point links.
- MACsec configuration is not supported on EtherChannel ports. Instead, MACsec configuration can be applied on the individual member ports of an EtherChannel. To remove MACsec configuration, you must first unbundle the member ports from the EtherChannel, and then remove it from the individual member ports.
- Host-to-switch MKA MACsec supports only 128-bit MACsec encryption for host to switch MACsec.
- Certificate-based MACsec is supported only if the access-session is configured as closed or in multiple-host mode. None of the other configuration modes are supported.
- Packet number exhaustion rekey is not supported.
- If the **dot1q tag vlan native** command is configured globally, the dot1x reauthentication will fail on trunk ports.
- MACsec XPN Cipher Suites do not provide confidentiality protection with a confidentiality offset, and these together are not supported in switch-to-switch MACsec connections.
- As per IEEE standards, the maximum value of replay window is 2^{30-1} for MACsec XPN Cipher Suites. Even if you configure a higher value than this, it will be restricted to 2^{30-1} only.
- MACsec with Precision Time Protocol (PTP) is not supported.
- MACsec switch-to-host connections in an overlay network are not supported.
- The **should-secure** access mode is supported on switch-to-switch ports only using PSK authentication.
- PSK fallback key chain is not supported on Ethernet Virtual Circuit (EVC) and point-to-multipoint cases.
- PSK fallback key chain supports infinite lifetime with one key only. The connectivity association key name (CKN) ID used in the fallback key chain must not match any of the CKN IDs used in the primary key chain.
- EAPOL packets of EtherType 0x888E are not intercepted by the interface if MACsec or dot1x is not enabled on the interface.
- If there are any intermediate switches present between two MACsec endpoints, any P2P protocols like STP or CDP will not work.
- Network-Based Application Recognition (NBAR) is not supported on MACsec switch-to-host connections.
- MACsec Cipher Announcement feature is not supported.
- Dot1x CTS with the Certificate based MACsec feature is not supported.
- Configuring custom EAPOL is not supported.

Information About MACsec Encryption

The following sections provide information about MACsec encryption.

Recommendations for MACsec Encryption

This section list the recommendations for configuring MACsec encryption:

- Use the confidentiality (encryption) offset as 0 in switch-to-host connections.
- Use Bidirectional Forwarding and Detection (BFD) timer value as 750 milliseconds for 10Gbps ports and 1.25 seconds for any port with speed above 10Gbps.
- Execute the **shutdown** command, and then the **no shutdown** command on a port, after changing any MKA policy or MACsec configuration for active sessions, so that the changes are applied to active sessions.
- Set the connectivity association key (CAK) rekey overlap timer to 30 seconds or more.
- Do not use Cisco TrustSec Security Association Protocol (SAP) MACsec encryption for port speeds above 10Gbps.
- Do not enable both Cisco TrustSec SAP and uplink MKA at the same time on any interface.
- Use MACsec MKA encryption.
- Do not use the **delay-protection** command when defining MKA policy if MACsec scale sessions are configured.

Note Whenever any change is performed on MKA policy or MACsec configuration on interface, it is mandatory to perform shutdown on interface and ensure that interface is down and then perform no shutdown on interface.

MACsec Encryption Overview

MACsec is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. Switches support 802.1AE encryption with MACsec Key Agreement (MKA) on switch-to-host links for encryption between the switch and host device. The switch also supports MACsec encryption for switch-to-switch (inter-network device) security using both Cisco TrustSec Network Device Admission Control (NDAC), Security Association Protocol (SAP) and MKA-based key exchange protocol.

Note When switch-to-switch MACSec is enabled, all traffic is encrypted, except the EAP-over-LAN (EAPOL) packets.

Link layer security can include both packet authentication between switches and MACsec encryption between switches (encryption is optional). Link layer security is supported on SAP-based MACsec.

Table 101: MACsec Support on Switch Ports

Connections	MACsec support
Switch-to-switch	MACsec MKA encryption (recommended) Cisco TrustSec SAP

Cisco TrustSec and Cisco SAP are meant only for switch-to-switch links and are not supported on switch ports connected to end hosts, such as PCs or IP phones. MKA is supported on switch-to-host facing links as well as switch-to-switch links. Host-facing links typically use flexible authentication ordering for handling heterogeneous devices with or without IEEE 802.1x, and can optionally use MKA-based MACsec encryption. Cisco NDAC and SAP are mutually exclusive with Network Edge Access Topology (NEAT), which is used for compact switches to extend security outside the wiring closet.

Media Access Control Security and MACsec Key Agreement

MACsec, defined in 802.1AE, provides MAC-layer encryption over wired networks by using out-of-band methods for encryption keying. The MACsec Key Agreement (MKA) Protocol provides the required session keys and manages the required encryption keys. MKA and MACsec are implemented after successful authentication using certificate-based MACsec or Pre Shared Key (PSK) framework.

A switch using MACsec accepts either MACsec or non-MACsec frames, depending on the policy associated with the MKA peer. MACsec frames are encrypted and protected with an integrity check value (ICV). When the switch receives frames from the MKA peer, it decrypts them and calculates the correct ICV by using session keys provided by MKA. The switch compares that ICV to the ICV within the frame. If they are not identical, the frame is dropped. The switch also encrypts and adds an ICV to any frames sent over the secured port (the access point used to provide the secure MAC service to a MKA peer) using the current session key.

The MKA Protocol manages the encryption keys used by the underlying MACsec protocol. The basic requirements of MKA are defined in 802.1x-REV. The MKA Protocol extends 802.1x to allow peer discovery with confirmation of mutual authentication and sharing of MACsec secret keys to protect data exchanged by the peers.

The EAP framework implements MKA as a newly defined EAP-over-LAN (EAPOL) packet. EAP authentication produces a master session key (MSK) shared by both partners in the data exchange. Entering the EAP session ID generates a secure connectivity association key name (CKN). The switch acts as the authenticator for both uplink and downlink; and acts as the key server for downlink. It generates a random secure association key (SAK), which is sent to the client partner. The client is never a key server and can only interact with a single MKA entity, the key server. After key derivation and generation, the switch sends periodic transports to the partner at a default interval of 2 seconds.

The packet body in an EAPOL Protocol Data Unit (PDU) is referred to as a MACsec Key Agreement PDU (MKPDU). MKA sessions and participants are deleted when the MKA lifetime (6 seconds) passes with no MKPDU received from a participant. For example, if a MKA peer disconnects, the participant on the switch continues to operate MKA until 6 seconds have elapsed after the last MKPDU is received from the MKA peer.

Note Integrity check value (ICV) indicator in MKPDU is optional. ICV is not optional when the traffic is encrypted.

EAPoL Announcements indicate the use of the type of keying material. The announcements can be used to announce the capability of the supplicant as well as the authenticator. Based on the capability of each side, the largest common denominator of the keying material could be used.

MKA Policies

To enable MKA on an interface, a defined MKA policy should be applied to the interface. You can configure these options:

- Policy name, not to exceed 16 ASCII characters.
- Confidentiality (encryption) offset of 0, 30, or 50 bytes for each physical interface

Definition of Policy-Map Actions

This section describes the policy-map actions and its definition:

- Activate: Applies a service template to the session.
- Authenticate: Starts authentication of the session.
- Authorize: Explicitly authorizes a session.
- Set-domain: Explicitly sets the domain of a client.
- Terminate: Terminates the method that is running, and deletes all the method details associated with the session.
- Deactivate: Removes the service-template applied to the session. If not applied, no action is taken.
- Set-timer: Starts a timer and gets associated with the session. When the timer expires, any action that needs to be started can be processed.
- Authentication-restart: Restarts authentication.
- Clear-session: Deletes a session.
- Pause: Pauses authentication.

Rest of the actions as self-explanatory and are associated with authentication.

Virtual Ports

Use virtual ports for multiple secured connectivity associations on a single physical port. Each connectivity association (pair) represents a virtual port. In uplink, you can have only one virtual port per physical port. You cannot simultaneously host secured and unsecured sessions in the same VLAN on the same port. Because of this limitation, 802.1x multiple authentication mode is not supported.

The exception to this limitation is in multiple-host mode when the first MACsec supplicant is successfully authenticated and connected to a hub that is connected to the switch. A non-MACsec host connected to the hub can send traffic without authentication because it is in multiple-host mode. We do not recommend using multi-host mode because after the first successful client, authentication is not required for other clients.

Virtual ports represent an arbitrary identifier for a connectivity association and have no meaning outside the MKA Protocol. A virtual port corresponds to a separate logical port ID. Valid port IDs for a virtual port are 0x0002 to 0xFFFF. Each virtual port receives a unique secure channel identifier (SCI) based on the MAC address of the physical interface concatenated with a 16-bit port ID.

MKA Statistics

Some MKA counters are aggregated globally, while others are updated both globally and per session. You can also obtain information about the status of MKA sessions. See Example: Displaying MKA Information, on page 1255 for further information.

Key Lifetime and Hitless Key Rollover

A MACsec key chain can have multiple pre-shared keys (PSK) each configured with a key id and an optional lifetime. A key lifetime specifies at which time the key expires. In the absence of a lifetime configuration, the default lifetime is unlimited. When a lifetime is configured, MKA rolls over to the next configured pre-shared key in the key chain after the lifetime is expired. Time zone of the key can be local or UTC. Default time zone is UTC.

You can Key rolls over to the next key within the same key chain by configuring a second key in the key chain and configuring a lifetime for the first key. When the lifetime of the first key expires, it automatically rolls over to the next key in the list. If the same key is configured on both sides of the link at the same time, then the key rollover is hitless, that is, key rolls over without traffic interruption.

On all participating devices, the MACsec key chain must be synchronised by using Network Time Protocol (NTP) and the same time zone must be used. If all the participating devices are not synchronized, the connectivity association key (CAK) rekey will not be initiated on all the devices at the same time.

Note The lifetime of the keys need to be overlapped in order to achieve hitless key rollover.

Fallback Key

The Fallback Key feature establishes an MKA session with the pre-shared fallback key whenever the primary pre-shared key (PSK) fails to establish a session because of key mismatch. This feature prevents downtime and ensures traffic hitless scenario during CAK mismatch (primary PSK) between the peers. The purpose of the fallback key chain is to act as a last resort key. The fallback key feature is only applicable for PSK based MKA or MACsec sessions.

Replay Protection Window Size

Replay protection is a feature provided by MACsec to counter replay attacks. Each encrypted packet is assigned a unique sequence number and the sequence is verified at the remote end. Frames transmitted through a Metro Ethernet service provider network are highly susceptible to reordering due to prioritization and load balancing mechanisms used within the network.

A replay window is necessary to support the use of MACsec over provider networks that reorder frames. Frames within the window can be received out of order, but are not replay protected. The default window size is 0, which enforces strict reception ordering. The replay window size can be configured in the range of 0 to 2^{32}- 1.

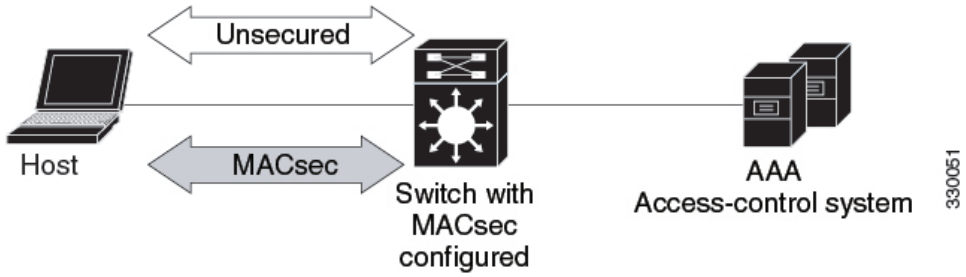
MACsec, MKA, and 802.1x Host Modes

You can use MACsec and the MKA Protocol with 802.1x single-host mode, multi-host mode, or Multi Domain Authentication (MDA) mode. Multiple authentication mode is not supported.

Single-Host Mode

The figure shows how a single EAP authenticated session is secured by MACsec by using MKA

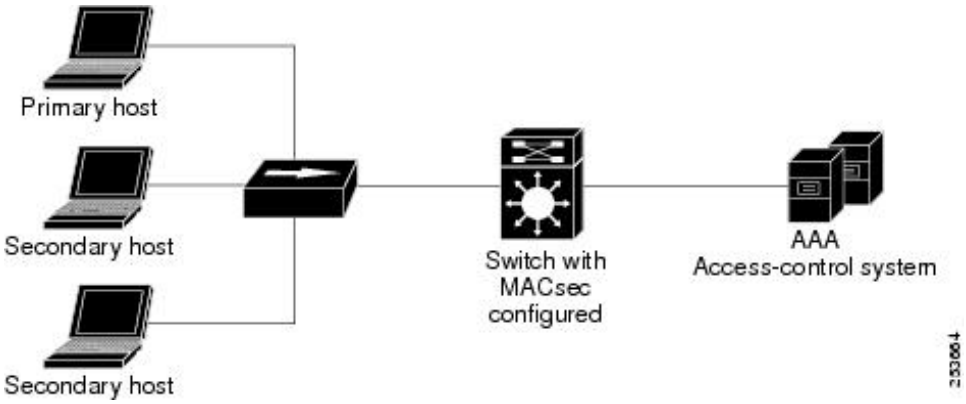
Figure 89: MACsec in Single-Host Mode with a Secured Data Session



Multiple Host Mode

In standard (not 802.1x REV) 802.1x multiple-host mode, a port is open or closed based on a single authentication. If one user, the primary secured client services client host, is authenticated, the same level of network access is provided to any host connected to the same port. If a secondary host is a MACsec supplicant, it cannot be authenticated and traffic would not flow. A secondary host that is a non-MACsec host can send traffic to the network without authentication because it is in multiple-host mode. The figure shows MACsec in Standard Multiple-Host Unsecure Mode.

Figure 90: MACsec in Multiple-Host Mode - Unsecured



Note Multi-host mode is not recommended because after the first successful client, authentication is not required for other clients, which is not secure.

Multiple-Domain Mode

In standard (not 802.1x REV) 802.1x multiple-domain mode, a port is open or closed based on a single authentication. If the primary user, a PC on data domain, is authenticated, the same level of network access is provided to any domain connected to the same port. If a secondary user is a MACsec supplicant, it cannot be authenticated and traffic would no flow. A secondary user, an IP phone on voice domain, that is a non-MACsec host, can send traffic to the network without authentication because it is in multiple-domain mode.

MACsec MKA using Certificate-based MACsec

MACsec MKA is supported on switch-to-switch links. Using certificate-based MACsec, you can configure MACsec MKA between device uplink ports. Certificate-based MACsec allows mutual authentication and obtains an MSK (master session key) from which the connectivity association key (CAK) is derived for MKA operations. Device certificates are carried, using certificate-based MACsec, for authentication to the AAA server.

Prerequisites for MACsec MKA using Certificate-based MACsec

- Ensure that you have a Certificate Authority (CA) server configured for your network.
- Generate a CA certificate.
- Ensure that you have configured Cisco Identity Services Engine (ISE) Release 2.0.
- Ensure that both the participating devices, the CA server, and Cisco Identity Services Engine (ISE) are synchronized using Network Time Protocol (NTP). If time is not synchronized on all your devices, certificates will not be validated.
- Ensure that 802.1x authentication and AAA are configured on your device.

MACsec Connection Across Intermediate Switches

The encrypted packets were dropped if WAN MACsec was configured on the end devices with MACsec not configured on the intermediate switches. With the ClearTag feature implemented on the ASIC, the switch forwards the encrypted packet without parsing the MACsec header.

Limitations for MACsec Connections Across Intermediate Switches

- Hop-by-hop MACsec encryption with switches as intermediate switches where WAN MACsec is configured on the routers is not supported.
- WAN MACsec configured on the routers with intermediate switches as the switches is not supported on Layer 3 VPNs.
- WAN MACsec configured on the routers with intermediate switches as the switches show CDP neighbors only in should-secure mode.

Switch-to-Switch MKA MACsec Must Secure Policy

Must-secure support is enabled on both the ingress and the egress. Must-secure is supported for MKA and SAP. With must-secure enabled, only EAPOL traffic will not be encrypted. The rest of the traffic will be encrypted. Unencrypted packets are dropped.

Note Must-secure mode is enabled by default.

With should-secure enabled, if the peer is configured for MACsec, the data traffic is encrypted, otherwise it is sent in clear text.

MKA/MACsec for Port Channel

MKA/MACsec can be configured on the port members of a port channel. MKA/MACsec is agnostic to the port channel since the MKA session is established between the port members of a port channel.

Note Etherchannel links that are formed as part of the port channel can either be congruent or disparate i.e. the links can either be MACsec-secured or non-MACsec-secured. MKA session between the port members is established even if a port member on one side of the port channel is not configured with MACsec.

It is recommended that you enable MKA/MACsec on all the member ports for better security of the port channel.

How to Configure MACsec Encryption

The following sections provide information about the various tasks that comprise MACsec encryption.

Configuring MKA and MACsec

MACsec is disabled by default. No MKA policies are configured.

Configuring an MKA Policy

Beginning in privileged EXEC mode, follow these steps to create an MKA Protocol policy. Note that MKA also requires that you enable 802.1x.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mka policy** *policy-name* **Example:** `Device(config)# mka policy mka_policy`	Identifies an MKA policy, and enters MKA policy configuration mode. The maximum policy name length is 16 characters. **Note** The default MACsec cipher suite in the MKA policy will always be "GCM-AES-128".

	Command or Action	Purpose
Step 4	**key-server** *priority* **Example:** `Device(config-mka-policy)# key-server priority 200`	Configures MKA key server options and set priority (between 0-255). **Note** When value of key server priority is set to 255, the peer can not become the key server. The key server priority value is valid only for MKA PSK; and not for MKA EAPTLS.
Step 5	**include-icv-indicator** **Example:** `Device(config-mka-policy)# include-icv-indicator`	Enables the ICV indicator in MKPDU. Use the **no** form of this command to disable the ICV indicator.
Step 6	**macsec-cipher-suite** *gcm-aes-128* **Example:** `Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128`	Configures a cipher suite for deriving SAK with 128-bit encryption.
Step 7	**confidentiality-offset** *offset-value* **Example:** `Device(config-mka-policy)# confidentiality-offset 0`	Set the confidentiality (encryption) offset for each physical interface. **Note** Offset Value can be 0, 30 or 50. If you are using Anyconnect on the client, it is recommended to use Offset 0.
Step 8	**ssci-based-on-sci** **Example:** `Device(config-mka-policy)# ssci-based-on-sci`	(Optional) Computes Short Secure Channel Identifier (SSCI) value based on Secure Channel Identifier (SCI) value. The higher the SCI value, the lower is the SSCI value.
Step 9	**end** **Example:** `Device(config-mka-policy)# end`	Exit enters MKA policy configuration mode and returns to privileged EXEC mode.
Step 10	**show mka policy** **Example:** `Device# show mka policy`	Displays MKA policy configuration information.

Configuring Switch-to-host MACsec Encryption

Follow these steps to configure MACsec on an interface with one MACsec session for voice and one for data:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter the password, if prompted.
Step 2	**configureterminal** **Example:** `Device>configure terminal`	Enters the global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Identifies the MACsec interface, and enters interface configuration mode. The interface must be a physical interface.
Step 4	**switchport access vlan***vlan-id* **Example:** `Device(config-if)# switchport access vlan 613`	Configures the access VLAN for the port.
Step 5	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Configures the interface as an access port.
Step 6	**macsec** **Example:** `Device(config-if)# macsec`	Enables 802.1ae MACsec on the interface. The macsec command enables MKA MACsec on switch-to-host links only.
Step 7	**access-session host-mode multi-domain** **Example:** `Device(config-if)# access-session host-mode multi-domain`	Configures authentication manager mode on the port to allow both a host and a voice device to be authenticated on the 802.1x-authorized port. If not configured, the default host mode is single.
Step 8	**access-session port-control auto** **Example:** `Device(config-if)# access-session port-control auto`	Enables 802.1x authentication on the port. The port changes to the authorized or unauthorized state based on the authentication exchange between the switch and the client.
Step 9	**authentication periodic** **Example:** `Device(config-if)# authentication periodic`	(Optional) Enables or disables re-authentication for this port .

	Command or Action	Purpose
Step 10	**authentication timer reauthenticate** **Example:** `Device(config-if)# authentication timer reauthenticate`	(Optional) Enters a value between 1 and 65535 (in seconds). Obtains re-authentication timeout value from the server. Default re-authentication time is 3600 seconds.
Step 11	**authentication violation protect** **Example:** `Device(config-if)# configure terminal`	Configures the port to drop unexpected incoming MAC addresses when a new device connects to a port or when a device connects to a port after the maximum number of devices are connected to that port. If not configured, the default is to shut down the port.
Step 12	**mka policy** *policy-name* **Example:** `Device(config-if)# mka policy mka_policy`	Applies an existing MKA protocol policy to the interface, and enable MKA on the interface. If no MKA policy was configured (by entering the**mka policy** global configuration command).
Step 13	**dot1x pae authenticator** **Example:** `Device(config-if)# dot1x pae authenticator`	Configures the port as an 802.1x port access entity (PAE) authenticator.
Step 14	**service-policy type control subscriber** *control-policy-name* **Example:** `Device(config-if)#service-policy type control subscriber DOT1X_POLICY_RADIUS`	Applies a previously configured control policy.
Step 15	**spanning-tree portfast** **Example:** `Device(config-if)# spanning-tree portfast`	Enables spanning tree Port Fast on the interface in all its associated VLANs. When the Port Fast feature is enabled, the interface changes directly from a blocking state to a forwarding state without making the intermediate spanning-tree state changes
Step 16	**end** **Example:** `Device(config)# end`	Exits interface configuration mode and returns to privileged EXEC mode. For more information on host-to-switch MACsec configuration examples, see Example: configuring host to switch MACsec, on page 1261.
Step 17	**show authentication session interface** *interface-id* **Example:** `Device# show authentication session interface GigabitEthernet 1/1`	Verifies the authorized session security status.

	Command or Action	Purpose
Step 18	**show mka sessions** **Example:** `Device# show mka sessions`	Verifies the established MKA sessions.

Configuring MACsec MKA using PSK

Beginning in privileged EXEC mode, follow these steps to configure MACsec MKA policies using a Pre Shared Key (PSK).

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**key chain** *key-chain-name* **macsec** **Example:** `Device(config)# key chain keychain1 macsec`	Configures a key chain and enters the key chain configuration mode.
Step 4	**key** *hex-string* **Example:** `Device(config-key-chain)# key 1000`	Configures a unique identifier for each key in the keychain and enters the keychain's key configuration mode. **Note** For 128-bit encryption, use any value between 1 and 32 hex digit key-string.
Step 5	**cryptographic-algorithm** *{aes-128-cmac | aes-256-cmac}* **Example:** `Device(config-key-chain)# cryptographic-algorithm aes-128-cmac`	Set cryptographic authentication algorithm with 128-bit or 256-bit encryption.
Step 6	**key-string** *{ [0|6|7] pwd-string | pwd-string}* **Example:** `Device(config-key-chain)# key-string 12345678901234567890123456789012`	Sets the password for a key string. Only hex characters must be entered.

	Command or Action	Purpose
Step 7	**lifetime local** [*start timestamp {hh::mm::ss | day | month | year}*] [**duration** *seconds* | *end timestamp {hh::mm::ss | day | month | year}*] **Example:** `Device(config-key-chain)# lifetime local 12:12:00 July 28 2016 12:19:00 July 28 2016`	Sets the lifetime of the pre shared key.
Step 8	**end** **Example:** `Device(config-key-chain)# end`	Exits key chain configuration mode and returns to privileged EXEC mode.

Configuring MACsec MKA on an Interface using PSK

Beginning in privileged EXEC mode, follow these steps to configure MACsec MKA policies on an interface using a Pre Shared Key (PSK).

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config-if)# interface GigabitEthernet 1/1`	Enters interface configuration mode.
Step 4	**macsec network-link** **Example:** `Device(config-if)# macsec network-link`	Enables MACsec on the interface.
Step 5	**mka policy** *policy-name* **Example:** `Device(config-if)# mka policy mka_policy`	Configures an MKA policy.
Step 6	**mka pre-shared-key key-chain** *key-chain name* [**fallback key-chain** *key-chain name*] **Example:**	Configures an MKA pre-shared-key key-chain name.

	Command or Action	Purpose
	`Device(config-if)# mka pre-shared-key key-chain key-chain-name`	
Step 7	**macsec replay-protection window-size** *frame number* **Example:** `Device(config-if)# macsec replay-protection window-size 10`	Sets the MACsec window size for replay protection.
Step 8	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

What to do next

It is not recommended to change the MKA policy on an interface with MKA PSK configured when the session is running. However, if a change is required, you must reconfigure the policy as follows:

1. Disable the existing session by removing macsec network-link configuration on each of the participating node using the **no macsec network-link** command
2. Configure the MKA policy on the interface on each of the participating node using the **mka policy policy-name** command.
3. Enable the new session on each of the participating node by using the **macsec network-link** command.

Configuring MACsec MKA using Certificate-based MACsec

To configure MACsec with MKA on point-to-point links, perform these tasks:

- Configure Certificate Enrollment
  - Generate Key Pairs
  - Configure SCEP Enrollment
  - Configure Certificates Manually
- Configure an Authentication Policy
- Configure certificate-based MACsec Profiles and IEEE 802.1x Credentials
- Configure MKA MACsec using certificate-based MACsec on Interfaces

Generating Key Pairs

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password, if prompted.
Step 2	**crypto key generate rsa label** *label-name* **general-keys modulus** *size* **Example:** `Device>crypto key generate rsa label general-keys  modulus 2048`	Generates a RSA key pair for signing and encryption. You can also assign a label to each key pair using the label keyword. The label is referenced by the trustpoint that uses the key pair. If you do not assign a label, the key pair is automatically labeled <Default-RSA-Key>. If you do not use additional keywords this command generates one general purpose RSA key pair. If the modulus is not specified, the default key modulus of 1024 is used. You can specify other modulus sizes with the modulus keyword.
Step 3	**show authentication session interface** *interface-id* **Example:** `Device# show authentication session interface gigabitethernet 1/1`	Verifies the authorized session security status.

Configuring Enrollment using SCEP

Simple Certificate Enrollment Protocol (SCEP) is a Cisco-developed enrollment protocol that uses HTTP to communicate with the certificate authority (CA) or registration authority (RA). SCEP is the most commonly used method for sending and receiving requests and certificates.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *server name* **Example:** `Device(config)# crypto pki trustpoint ka`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.

	Command or Action	Purpose
Step 4	**enrollment url** *url name pem* **Example:** `Device(ca-trustpoint)# enrollment url http://url:80`	Specifies the URL of the CA on which your device should send certificate requests. An IPv6 address can be added in the URL enclosed in brackets. For example: http://[2001:DB8:1:1::1]:80. The pem keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request.
Step 5	**rsakeypair** *label* **Example:** `Device(ca-trustpoint)# rsakeypair exampleCAkeys`	Specifies which key pair to associate with the certificate. **Note** The **rsakeypair** name must match the trust-point name.
Step 6	**serial-number none** **Example:** `Device(ca-trustpoint)# serial-number none`	The **none** keyword specifies that a serial number will not be included in the certificate request.
Step 7	**ip-address none** **Example:** `Device(ca-trustpoint)# ip-address none`	The **none** keyword specifies that no IP address should be included in the certificate request.
Step 8	**revocation-check crl** **Example:** `Device(ca-trustpoint)# revocation-check crl`	Specifies CRL as the method to ensure that the certificate of a peer has not been revoked.
Step 9	**auto-enroll** *percent* **regenerate** **Example:** `Device(ca-trustpoint)# auto-enroll 90 regenerate`	Enables auto-enrollment, allowing the client to automatically request a rollover certificate from the CA. If auto-enrollment is not enabled, the client must be manually re-enrolled in your PKI upon certificate expiration. By default, only the Domain Name System (DNS) name of the device is included in the certificate. Use the percent argument to specify that a new certificate will be requested after the percentage of the lifetime of the current certificate is reached. Use the regenerate keyword to generate a new key for the certificate even if a named key already exists.

	Command or Action	Purpose
		If the key pair being rolled over is exportable, the new key pair will also be exportable. The following comment will appear in the trustpoint configuration to indicate whether the key pair is exportable: "! RSA key pair associated with trustpoint is exportable." It is recommended that a new key pair be generated for security reasons.
Step 10	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 11	**crypto pki authenticate** *name* **Example:** `Device(config)# crypto pki authenticate myca`	Retrieves the CA certificate and authenticates it.
Step 12	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 13	**show crypto pki certificate** *trustpoint name* **Example:** `Device# show crypto pki certificate ka`	Displays information about the certificate for the trust point.

Configuring Enrollment Manually

If your CA does not support SCEP or if a network connection between the router and CA is not possible. Perform the following task to set up manual certificate enrollment:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *server name* **Example:**	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.

	Command or Action	Purpose
	`Device# crypto pki trustpoint ka`	
Step 4	**enrollment url** *url name pem* **Example:** `Device(ca-trustpoint)# enrollment url http://url:80`	Specifies the URL of the CA on which your device should send certificate requests. An IPv6 address can be added in the URL enclosed in brackets. For example: http://[2001:DB8:1:1::1]:80. The pem keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request.
Step 5	**rsakeypair** *label* **Example:** `Device(ca-trustpoint)# rsakeypair exampleCAkeys`	Specifies which key pair to associate with the certificate.
Step 6	**serial-number none** **Example:** `Device(ca-trustpoint)# serial-number none`	The **none** keyword specifies that a serial number will not be included in the certificate request.
Step 7	**ip-address none** **Example:** `Device(ca-trustpoint)# ip-address none`	The **none** keyword specifies that no IP address should be included in the certificate request.
Step 8	**revocation-check crl** **Example:** `Device(ca-trustpoint)# revocation-check crl`	Specifies CRL as the method to ensure that the certificate of a peer has not been revoked.
Step 9	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 10	**crypto pki authenticate** *name* **Example:** `Device(config)# crypto pki authenticate myca`	Retrieves the CA certificate and authenticates it.
Step 11	**crypto pki enroll** *name* **Example:** `Device(config)# crypto pki enroll myca`	Generates certificate request and displays the request for copying and pasting into the certificate server. Enter enrollment information when you are prompted. For example, specify whether to include the device FQDN and IP address in the certificate request.

	Command or Action	Purpose
		You are also given the choice about displaying the certificate request to the console terminal. The base-64 encoded certificate with or without PEM headers as requested is displayed.
Step 12	**crypto pki import** *name* **certificate** **Example:** `Device(config)# crypto pki import myca certificate`	Imports a certificate via TFTP at the console terminal, which retrieves the granted certificate. The device attempts to retrieve the granted certificate via TFTP using the same filename used to send the request, except the extension is changed from ".req" to ".crt". For usage key certificates, the extensions "-sign.crt" and "-encr.crt" are used. The device parses the received files, verifies the certificates, and inserts the certificates into the internal certificate database on the switch. **Note** Some CAs ignore the usage key information in the certificate request and issue general purpose usage certificates. If your CA ignores the usage key information in the certificate request, only import the general purpose certificate. The router will not use one of the two key pairs generated.
Step 13	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 14	**show crypto pki certificate** *trustpoint name* **Example:** `Device# show crypto pki certificate ka`	Displays information about the certificate for the trust point.

Configuring switch-to-switch MACsec encryption

To apply MACsec MKA using certificate-based MACsec to interfaces, perform the following task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Identifies the MACsec interface, and enters interface configuration mode. The interface must be a physical interface.
Step 4	**macsec network-link** **Example:** `Device(config-if)# macsec network-link`	Enables MACsec on the interface.
Step 5	**authentication periodic** **Example:** `Device(config-if)# authentication periodic`	Enables reauthentication for this port.
Step 6	**authentication timer reauthenticate interval** **Example:** `Device(config-if)# authentication timer reauthenticate interval`	Sets the reauthentication interval.
Step 7	**authentication host-mode multi-host** **Example:** `Device(config-if)# authentication host-mode multi-host`	Allows hosts to gain access to the interface.
Step 8	**access-session closed** **Example:** `Device(config-if)# access-session closed`	Prevents preauthentication access on the interface.
Step 9	**access-session port-control auto** **Example:** `Device(config-if)# access-session port-control auto`	Sets the authorization state of a port.
Step 10	**dot1x pae both** **Example:** `Device(config-if)# dot1x pae both`	Configures the port as an 802.1X port access entity (PAE) supplicant and authenticator.
Step 11	**service-policy type control subscriber** *control-policy-name* **Example:** `Device(config-if)#service-policy type control subscriber DOT1X_POLICY_RADIUS`	Applies a previously configured control policy.

	Command or Action	Purpose
Step 12	**dot1x credentials profile** **Example:** `Device(config-if)# dot1x credentials profile`	Assigns a 802.1x credentials profile to the interface.
Step 13	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mdoe and returns to privileged EXEC mode. For more information on switch-to-switch MACsec configuration examples, see Example: configure multi-domain, on page 1262.
Step 14	**show macsec interface** *interface-id* **Example:** `Device# show macsec interface GigabitEthernet 1/1`	Displays MACsec details for the interface.

Configuring MKA/MACsec for Port Channel using PSK

Beginning in privileged EXEC mode, follow these steps to configure MKA policies on an interface using a Pre Shared Key (PSK).

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config-if)# interface gigabitethernet 1/1`	Enters interface configuration mode.
Step 4	**macsec network-link** **Example:** `Device(config-if)# macsec network-link`	Enables MACsec on the interface. Supports layer 2 and layer 3 port channels.
Step 5	**mka policy** *policy-name* **Example:** `Device(config-if)# mka policy mka_policy`	Configures an MKA policy.

	Command or Action	Purpose
Step 6	**mka pre-shared-key key-chain** *key-chain name* [**fallback key-chain** *key-chain name*] **Example:** `Device(config-if)# mka pre-shared-key key-chain key-chain-name`	Configures an MKA pre-shared-key key-chain name. **Note** The MKA pre-shared key can be configured on either physical interface or sub-interfaces and not on both.
Step 7	**macsec replay-protection window-size** *frame number* **Example:** `Device(config-if)# macsec replay-protection window-size 0`	Sets the MACsec window size for replay protection.
Step 8	**channel-group** *channel-group-number* **mode** {**active** | **passive**} **Example:** `Device(config-if)# channel-group 3 mode active`	Configures the port in a channel group and sets the mode. **Note** You cannot configure ports in a channel group without configuring MACsec on the interface. You must configure the commands in Step 3, 4, 5 and 6 before this step. The channel-number range is from 1 to 4096. The port channel associated with this channel group is automatically created if the port channel does not already exist.For mode, select one of the following keywords: • **active** — Enables LACP only if a LACP device is detected. It places the port into an active negotiating state in which the port starts negotiations with other ports by sending LACP packets. • **passive** — Enables LACP on the port and places it into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation.
Step 9	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Port Channel Logical Interfaces for Layer 2 EtherChannels

To create a port channel interface for a Layer 2 EtherChannel, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-group-number* **Example:** `Device(config)# interface port-channel 1`	Creates the port channel interface. **Note** Use the **no** form of this command to delete the port channel interface.
Step 4	**switchport** **Example:** `Device(config-if)# switchport`	Switches an interface that is in Layer 3 mode into Layer 2 mode for Layer 2 configuration.
Step 5	**switchport mode** {**access** | **trunk**} **Example:** `Device(config-if)# switchport mode access`	Assigns all ports as static-access ports in the same VLAN, or configure them as trunks.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Port Channel Logical Interfaces for Layer 3 EtherChannels

To create a port channel interface for a Layer 3 EtherChannel, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode.
Step 4	**no switchport** **Example:** `Device(config-if)# no switchport`	Switches an interface that is in Layer 2 mode into Layer 3 mode for Layer 3 configuration.
Step 5	**ip address** *ip-address subnet_mask* **Example:** `Device(config-if)# ip address 192.0.2.10 255.255.255.254`	Assigns an IP address and subnet mask to the EtherChannel.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Cisco TrustSec MACsec

Configuring Cisco TrustSec Switch-to-Switch Link Security in Manual Mode

Before you begin

When manually configuring Cisco TrustSec on an interface, consider these usage guidelines and restrictions:

- If no SAP parameters are defined, Cisco TrustSec encapsulation or encryption is not performed.
- If you select GCM as the SAP operating mode, you must have a MACsec Encryption software license from Cisco. If you select GCM without the required license, the interface is forced to a link-down state.
- These protection levels are supported when you configure SAP pairwise master key (sap pmk):
  - SAP is not configured: no protection.
  - **sap mode-list gcm-encrypt gmac no-encap**: protection desirable but not mandatory.
  - **sap mode-list gcm-encrypt gmac**: confidentiality preferred and integrity required. The protection is selected by the supplicant according to supplicant preference.
  - **sap mode-list gmac**: integrity only.
  - **sap mode-list gcm-encrypt**: confidentiality required.
  - **sap mode-list gmac gcm-encrypt**: integrity required and preferred, confidentiality optional.
- Before changing the configuration from MKA to Cisco TrustSec SAP and vice versa, we recommend that you remove the interface configuration.

Beginning in privileged EXEC mode, follow these steps to manually configure Cisco TrustSec on an interface to another Cisco TrustSec device:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	**Note** Enters interface configuration mode.
Step 3	**cts manual** **Example:** `Device(config-if)# cts manual`	Enters Cisco TrustSec manual configuration mode.
Step 4	**sap pmk** *key* [**mode-list** *mode1* [*mode2* [*mode3* [*mode4*]]]] **Example:** `Device(config-if-cts-manual)# sap pmk 1234abcdef mode-list gcm-encrypt null no-encap`	(Optional) Configures the SAP pairwise master key (PMK) and operation mode. SAP is disabled by default in Cisco TrustSec manual mode. • *key*: A hexadecimal value with an even number of characters and a maximum length of 32 characters. The SAP operation mode options: • **gcm-encrypt**: Authentication and encryption **Note** Select this mode for MACsec authentication and encryption if your software license supports MACsec encryption. • **gmac**: Authentication, no encryption • **no-encap**: No encapsulation • **null**: Encapsulation, no authentication or encryption **Note** If the interface is not capable of data link encryption, **no-encap** is the default and the only available SAP operating mode. SGT is not supported.

	Command or Action	Purpose
Step 5	**no propagate sgt** **Example:** `Device(config-if-cts-manual)# no propagate sgt`	Use the **no** form of this command when the peer is incapable of processing a SGT. The **no propagate sgt** command prevents the interface from transmitting the SGT to the peer.
Step 6	**exit** **Example:** `Device(config-if-cts-manual)# exit`	Exits Cisco TrustSec 802.1x interface configuration mode.
Step 7	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 8	**show cts interface** [*interface-id* | **brief** | **summary**]	(Optional) Verify the configuration by displaying TrustSec-related interface characteristics.
Step 9	**copy running-config startup-config** **Example:** `Device#copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Examples for MACsec Encryption

Example: Configuring MKA and MACsec

This example shows how to create an MKA policy:

```
Device> enable
Device# configure terminal
Device(config)# mka policy mka_policy
Device(config-mka-policy)# key-server priority 200
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 30
Device(config-mka-policy)# ssci-based-on-sci
Device(config-mka-policy)#end
```

Examples: Configuring MACsec MKA using PSK

This example shows how to configure MACsec MKA using PSK.

```
Device> enable
Device# configure terminal
Device(config)# Key chain keychain1 macsec
Device(config-keychain)# key 1000
Device(config-keychain-key)# cryptographic-algorithm aes-128-cmac
Device(config-keychain-key)# key-string 12345678901234567890123456789012
```

```
Device(config-keychain-key)# lifetime local 12:12:00 July 28 2016 12:19:00 July 28 2016
Device(config-keychain-key)# end
```

This example shows how to configure MACsec MKA on an interface using PSK.

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 1/1
Device(config-if)# mka policy mka_policy
Device(config-if)# mka pre-shared-key key-chain key-chain-name
Device(config-if)# macsec replay-protection window-size 10
Device(config-if)# end
```

Examples: Configuring MACsec MKA using Certificate-based MACsec Remote Authentication

This example shows how to configure MACsec MKA using certificate-based MACsec remote authentication:

```
Device> enable
Device# configure terminal
Device(config)#aaa new-model
Device(config)#dot1x system-auth-control
Device(config)#radius server ISE
Device(config)#address ipv4 <ISE ipv4 address> auth-port 1645 acct-port 1646
Device(config)#automate-tester username cisco
Device(config)#key c1sco123
Device(config)#radius-server deadtime 2
!
Device(config)#aaa group server radius ISEGRP
Device(config)#server name ISE
!
Device(config)#aaa authentication dot1x default group ISEGRP
Device(config)#aaa authorization network default group ISEGRP
Device(config)#policy-map type control subscriber DOT1X_POLICY_RADIUS
Device(config)#event session-started match-all
Device(config)#10 class always do-until-failure
Device(config)#10 authenticate using dot1x both
Device(config)#event authentication-failure match-all
Device(config)#10 class always do-until-failure
Device(config)#10 terminate dot1x
Device(config)#20 authentication-restart 10
Device(config)#eap profile EAPTLS-PROF-IOSCA
Device(config)#method tls
Device(config)#pki-trustpoint POLESTAR-IOS-CA
!
Device(config)#dot1x credentials EAPTLSCRED-IOSCA
Device(config)#username xyz@polestar.cisco.com
Device(config)#pki-trustpoint POLESTAR-IOS-CA
Device(config-if)#interface GigabitEthernet1/1
Device(config-if)#macsec network-link
Device(config-if)#authentication periodic
Device(config-if)#authentication timer reauthenticate <reauthentication interval>
Device(config-if)#access-session host-mode multi-host
Device(config-if)#access-session closed
Device(config-if)#access-session port-control auto
Device(config-if)#dot1x pae both
Device(config-if)#dot1x credentials EAPTLSCRED-IOSCA
Device(config-if)#dot1x supplicant eap profile EAPTLS-PROF-IOSCA
Device(config-if)#service-policy type control subscriber DOT1X_POLICY_RADIUS
Device(config-if)# end
```

Examples: Configuring MACsec MKA using Certificate-based MACsec Local Authentication

Etherchannel Mode — Static/On

This example shows how to configure MACsec MKA using certificate-based MACsec remote authentication:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# dot1x system-auth-control
!
Device(config)# aaa authentication dot1x default local
Device(config)#aaa authorization network default local
Device(config)#aaa authorization credential-download default local
Device(config)#aaa local authentication default authorization default
Device(config)#aaa attribute list MUSTS-CA
Device(config)#attribute type linksec-policy must-secure
Device(config)#username MUST aaa attribute list MUSTS-CA
Device(config)#dot1x credentials MUSTS
Device(config)#username MUST
Device(config)#policy-map type control subscriber DOT1X_POLICY_LOCAL
Device(config)#event session-started match-all
Device(config)#10 class always do-until-failure
Device(config)#10 authenticate using dot1x both
Device(config)#event authentication-failure match-all
Device(config)#10 class always do-until-failure
Device(config)#10 terminate dot1x
Device(config)#20 authentication-restart 10
Device(config)#eap profile EAPTLS-PROF-IOSCA
Device(config)#method tls
Device(config)#pki-trustpoint POLESTAR-IOS-CA
!
Device(config-if)#interface GigabitEthernet1/1
Device(config-if)#macsec network-link
Device(config-if)#authentication periodic
Device(config-if)#authentication timer reauthenticate <reauthentication interval>
Device(config-if)#access-session host-mode multi-host
Device(config-if)#access-session closed
Device(config-if)#access-session port-control auto
Device(config-if)#dot1x pae both
Device(config-if)#dot1x credentials MUSTS
Device(config-if)#dot1x authenticator eap profile EAPTLS-PROF-IOSCA
Device(config-if)#dot1x supplicant eap profile EAPTLS-PROF-IOSCA
Device(config-if)#service-policy type control subscriber DOT1X_POLICY_RADIUS
Device(config-if)# end
```

Examples: Configuring MACsec MKA using Certificate-based MACsec Fallback Local Authentication

This example shows how to configure MACsec MKA using certificate-based MACsec fallback local authentication:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# dot1x system-auth-control
Device(config)# radius server ISE
```

```
Device(config)# address ipv4 <ISE ipv4 address> auth-port 1645 acct-port 1646
Device(config)# automate-tester username cisco
Device(config)# key c1sco123
Device(config)# radius-server deadtime 2
!
Device(config)#aaa group server radius ISEGRP
Device(config)#server name ISE
!
Device(config)#aaa authorization credential-download default group ISEGRP local
Device(config)#aaa local authentication default authorization default
Device(config)#aaa attribute list MUSTS-CA
Device(config)#attribute type linksec-policy must-secure
Device(config)#username MUST aaa attribute list MUSTS-CA
Device(config)#dot1x credentials MUSTS
Device(config)#username MUST
Device(config)#aaa authorization network default group ISEGRP local
Device(config)#policy-map type control subscriber DOT1X_POLICY_RADIUS
Device(config)#event session-started match-all
Device(config)#10 class always do-until-failure
Device(config)#10 authenticate using dot1x both
Device(config)#event authentication-failure match-all
Device(config)#10 class always do-until-failure
Device(config)#10 terminate dot1x
Device(config)#20 authentication-restart 10
Device(config)#eap profile EAPTLS-PROF-IOSCA
Device(config)#method tls
Device(config)#pki-trustpoint POLESTAR-IOS-CA
!
Device(config-if)#interface GigabitEthernet1/1
Device(config-if)#macsec network-link
Device(config-if)#authentication periodic
Device(config-if)#authentication timer reauthenticate <reauthentication interval>
Device(config-if)#access-session host-mode multi-host
Device(config-if)#access-session closed
Device(config-if)#access-session port-control auto
Device(config-if)#dot1x pae both
Device(config-if)#dot1x credentials MUSTS
Device(config-if)#dot1x authenticator eap profile EAPTLS-PROF-IOSCA
Device(config-if)#dot1x supplicant eap profile EAPTLS-PROF-IOSCA
Device(config-if)#service-policy type control subscriber DOT1X_POLICY_RADIUS
Device(config-if)# end
```

Example: Configuring MACsec MKA for Port Channel using PSK

Etherchannel Mode — Static/On

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode on:

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
```

```
Device(config)# interface gigabitethernet 1/1
Device(config-if)# channel-group 2 mode on
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/2
Device(config-if)# channel-group 2 mode on
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the**show etherchannel summary** command:

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------
```

```
2       Po2(RU)           -         ge1/1(P)   ge1/2(P)
```

Layer 3 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 192.0.2.10 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 192.0.2.11 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

2       Po2(RU)           -         ge1/1(P)   ge1/2(P)
```

Etherchannel Mode — LACP

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode as LACP.

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
```

```
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/1
Device(config-if)# channel-group 2 mode active
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/1
Device(config-if)# channel-group 2 mode active
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 1
Number of aggregators:           1
```

```
------+------------+-----------+-----------------------------------------------

 2      Po2(SU)          LACP      ge1/1(P)   ge1/2(P)
```

Layer 3 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 192.0.2.12 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 192.0.2.13 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
 Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+------------+-----------+-----------------------------------------------

 2      Po2(RU)          LACP      ge1/1(P)   ge1/2(P)
```

Etherchannel Mode — PAgP

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode as PAgP:

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
```

```
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/1
Device(config-if)# channel-group 2 mode desirable
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/1
Device(config-if)# channel-group 2 mode desirable
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

The following shows a sample output from the **show etherchannel summary** command.

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 1
```

```
 Number of aggregators:           1

------+------------+----------+-----------------------------------------------

 2      Po2(SU)          PAgP      Te1/1(P)   Te1/1(P)
```

Layer 3 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 192.0.2.12 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 192.0.2.13 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
 Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+------------+----------+-----------------------------------------------

 2      Po2(RU)          PAgP      ge1/1(P)   ge1/2(P)
```

Displaying Active MKA Sessions

The following shows all the active MKA sessions.

```
Device# show mka sessions interface ge1/1

=======================================================================================
Interface      Local-TxSCI          Policy-Name      Inherited
Key-Server
Port-ID        Peer-RxSCI           MACsec-Peers     Status             CKN
=======================================================================================

ge1/1          00a3.d144.3364/0025 POLICY           NO                  NO

37             701f.539b.b0c6/0032 1                Secured
1000
```

Example: Displaying MKA Information

The following is a sample output from the **show mka sessions** command:

```
Device# show mka sessions

Total MKA Sessions....... 1
      Secured Sessions... 1
      Pending Sessions... 0

=============================================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server
Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN
=============================================================================================================
Gi1/1        204c.9e85.ede4/002b p2               NO                YES
43             c800.8459.e764/002a 1                Secured
01000000000000000000000000000000000000000000000000000000000000000000000000000000
```

The following is a sample output from the **show mka sessions interface** *interface-name* command:

```
Device# show mka sessions interface GigabitEthernet 1/1

Summary of All Currently Active MKA Sessions on Interface GigabitEthernet1/1...

=============================================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server
Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN
=============================================================================================================
Gi1/1        204c.9e85.ede4/002b p2               NO                YES
43             c800.8459.e764/002a 1                Secured
01000000000000000000000000000000000000000000000000000000000000000000000000000000
```

The following is sample output from the **show mka sessions interface** *interface-name* **detail** command.

```
Device# show mka sessions interface GigabitEthernet 1/1 detail

MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec

Local Tx-SCI.............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/1
Audit Session ID.........
CAK Name (CKN)........... 01000000000000000000000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89567
```

```
EAP Role.................. NA
Key Server................ YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............. 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (1)
Old SAK Status............ FIRST-SAK
Old SAK AN................ 0
Old SAK KI (KN).......... FIRST-SAK (0)

SAK Transmit Wait Time... 0s (Not waiting for any peers to respond)
SAK Retire Time.......... 0s (No Old SAK to retire)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000001 (GCM-AES-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

# of MACsec Capable Live Peers............ 1
# of MACsec Capable Live Peers Responded.. 1

Live Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89555       c800.8459.e764/002a   10

Potential Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------

Dormant Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
```

The following is a sample output from the **show mka sessions details** command:

```
Device# show mka sessions details

MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec

Local Tx-SCI............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/1
Audit Session ID.........
CAK Name (CKN)........... 01000000000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89572
EAP Role.................. NA
Key Server................ YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............. 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (1)
```

```
Old SAK Status..... FIRST-SAK
Old SAK AN..... 0
Old SAK KI (KN)..... FIRST-SAK (0)

SAK Transmit Wait Time... 0s (Not waiting for any peers to respond)
SAK Retire Time..... 0s (No Old SAK to retire)

MKA Policy Name..... p2
Key Server Priority..... 2
Delay Protection..... NO
Replay Protection..... YES
Replay Window Size..... 0
Confidentiality Offset... 0
Algorithm Agility..... 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite..... 0080C20001000001 (GCM-AES-128)
MACsec Capability..... 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired..... YES

# of MACsec Capable Live Peers..... 1
# of MACsec Capable Live Peers Responded.. 1

Live Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89560       c800.8459.e764/002a   10

Potential Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------

Dormant Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
```

The following is a sample output from the **show mka policy** command:

```
Device# show mka policy

MKA Policy Summary...

Policy            KS       Delay   Replay  Window Conf   Cipher          Interfaces
Name              Priority Protect Protect Size   Offset Suite(s)        Applied
=============================================================================================
*DEFAULT POLICY*  0        FALSE   TRUE    0      0      GCM-AES-128

p1                1        FALSE   TRUE    0      0      GCM-AES-128

p2                2        FALSE   TRUE    0      0      GCM-AES-128     Gi1/1
```

The following is a sample output from the **show mka policy** *policy-name* command:

```
Device# show mka policy p2

MKA Policy Summary...

Policy            KS       Delay   Replay  Window Conf   Cipher          Interfaces
Name              Priority Protect Protect Size   Offset Suite(s)        Applied
=============================================================================================
p2                2        FALSE   TRUE    0      0      GCM-AES-128     Gi1/1
```

The following is a sample output from the **show mka policy** *policy-name* **detail** command:

```
Device# show mka policy p2 detail

MKA Policy Configuration ("p2")
```

```
========================
MKA Policy Name........ p2
Key Server Priority.... 2
Confidentiality Offset. 0
Send Secure Announcement..DISABLED
Cipher Suite(s)........ GCM-AES-128

Applied Interfaces...
  GigabitEthernet1/1
```

The following is a sample output from the **show mka statistics interface** *interface-name* command:

```
Device# show mka statistics interface GigabitEthernet 1/1

MKA Statistics for Session
==========================
Reauthentication Attempts.. 0

CA Statistics
   Pairwise CAKs Derived... 0
   Pairwise CAK Rekeys..... 0
   Group CAKs Generated.... 0
   Group CAKs Received..... 0

SA Statistics
   SAKs Generated.......... 1
   SAKs Rekeyed............ 0
   SAKs Received........... 0
   SAK Responses Received.. 1

MKPDU Statistics
   MKPDUs Validated & Rx... 89585
      "Distributed SAK".. 0
      "Distributed CAK".. 0
   MKPDUs Transmitted...... 89596
      "Distributed SAK".. 1
      "Distributed CAK".. 0
```

The following is a sample output from the **show mka summary** command:

```
Device# show mka summary

Total MKA Sessions....... 1
      Secured Sessions... 1
      Pending Sessions... 0

================================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server
Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN
================================================================================================
Gi1/1        204c.9e85.ede4/002b p2               NO                YES
43             c800.8459.e764/002a 1                Secured
01000000000000000000000000000000000000000000000000000000000000000000



MKA Global Statistics
=====================
MKA Session Totals
   Secured..................... 1
   Reauthentication Attempts.. 0

   Deleted (Secured).......... 0
   Keepalive Timeouts......... 0
```

```
CA Statistics
   Pairwise CAKs Derived...... 0
   Pairwise CAK Rekeys........ 0
   Group CAKs Generated....... 0
   Group CAKs Received........ 0

SA Statistics
   SAKs Generated............. 1
   SAKs Rekeyed............... 0
   SAKs Received.............. 0
   SAK Responses Received..... 1

MKPDU Statistics
   MKPDUs Validated & Rx...... 89589
      "Distributed SAK"..... 0
      "Distributed CAK"..... 0
   MKPDUs Transmitted......... 89600
      "Distributed SAK"..... 1
      "Distributed CAK"..... 0

MKA Error Counter Totals
========================
Session Failures
   Bring-up Failures.............. 0
   Reauthentication Failures........ 0
   Duplicate Auth-Mgr Handle........ 0

SAK Failures
   SAK Generation................... 0
   Hash Key Generation.............. 0
   SAK Encryption/Wrap.............. 0
   SAK Decryption/Unwrap............ 0
   SAK Cipher Mismatch.............. 0

CA Failures
   Group CAK Generation............. 0
   Group CAK Encryption/Wrap........ 0
   Group CAK Decryption/Unwrap...... 0
   Pairwise CAK Derivation.......... 0
   CKN Derivation................... 0
   ICK Derivation................... 0
   KEK Derivation................... 0
   Invalid Peer MACsec Capability... 0
MACsec Failures
   Rx SC Creation................... 0
   Tx SC Creation................... 0
   Rx SA Installation............... 0
   Tx SA Installation............... 0

MKPDU Failures
   MKPDU Tx......................... 0
   MKPDU Rx Validation.............. 0
   MKPDU Rx Bad Peer MN............. 0
   MKPDU Rx Non-recent Peerlist MN.. 0
```

The following is a sample output from the **show macsec interface** command:

```
Device# show macsec interface Gi 1/1

 MACsec is enabled
  Replay protect : enabled
  Replay window : 0
  Include SCI : yes
  Use ES Enable : no
```

```
 Use SCB Enable : no
 Admin Pt2Pt MAC : forceTrue(1)
 Pt2Pt MAC Operational : no
 Cipher : GCM-AES-128
 Confidentiality Offset : 0

Capabilities
 ICV length : 16
 Data length change supported: yes
 Max. Rx SA : 16
 Max. Tx SA : 16
 Max. Rx SC : 8
 Max. Tx SC : 8
 Validate Frames : strict
 PN threshold notification support : Yes
 Ciphers supported : GCM-AES-128
                     GCM-AES-256
                     GCM-AES-XPN-128
                     GCM-AES-XPN-256

Access control : must secure

Transmit Secure Channels
 SCI : 3C5731BBB5850475
 SC state : inUse(1)
  Elapsed time : 7w0d
  Start time : 7w0d
  Current AN: 0
  Previous AN: -
  Next PN: 149757
  SA State: inUse(1)
  Confidentiality : yes
  SAK Unchanged : yes
  SA Create time : 00:04:41
  SA Start time : 7w0d
  SC Statistics
   Auth-only Pkts : 0
   Auth-only Bytes : 0
   Encrypted Pkts : 0
   Encrypted Bytes : 0
  SA Statistics
   Auth-only Pkts : 0
   Auth-only Bytes : 0
   Encrypted Pkts : 149756
   Encrypted Bytes : 16595088

 Port Statistics
  Egress untag pkts  0
  Egress long pkts  0

Receive Secure Channels
 SCI : 3C5731BBB5C504DF
 SC state : inUse(1)
  Elapsed time : 7w0d
  Start time : 7w0d
  Current AN: 0
  Previous AN: -
  Next PN: 149786
  RX SA Count: 0
  SA State: inUse(1)
  SAK Unchanged : yes
  SA Create time : 00:04:39
  SA Start time : 7w0d
  SC Statistics
```

```
    Notvalid pkts 0
    Invalid pkts 0
    Valid pkts 0
    Late pkts 0
    Uncheck pkts 0
    Delay pkts 0
    UnusedSA pkts 0
    NousingSA pkts 0
    Validated Bytes 0
    Decrypted Bytes 0
   SA Statistics
    Notvalid pkts 0
    Invalid pkts 0
    Valid pkts 149784
    Late pkts 0
    Uncheck pkts 0
    Delay pkts 0
    UnusedSA pkts 0
    NousingSA pkts 0
    Validated Bytes 0
    Decrypted Bytes 16654544

  Port Statistics
   Ingress untag pkts  0
   Ingress notag pkts  631726
   Ingress badtag pkts  0
   Ingress unknownSCI pkts  0
   Ingress noSCI pkts  0
   Ingress overrun pkts  0
```

Example: configuring host to switch MACsec

This example shows how to configure host to switch MACsec:

```
Device>enable
Device#configure terminal
Device(config)#aaa new-model
!
Device(config)#aaa group server radius tests
Device(config)#server name RAD-1
!
Device(config)#aaa authentication dot1x default group tests
Device(config)#aaa authorization network default group tests
!
Device(config)#dot1x system-auth-control
!
Device(config)#policy-map type control subscriber TEST
Device(config)#event session-started match-all
Device(config)#10 class always do-until-failure
Device(config)#10 authenticate using dot1x priority 10
!
Device(config-if)#interface GigabitEthernet1/2
Device(config-if)#macsec
Device(config-if)#switchport access vlan 613
Device(config-if)#switchport mode access
Device(config-if)#access-session host-mode single-host
Device(config-if)#access-session port-control auto
Device(config-if)#dot1x pae authenticator
Device(config-if)#service-policy type control subscriber TEST
!
Device(config-if)#radius server RAD-1
```

```
Device(config-if)#address ipv4 <ISE ipv4 address> auth-port 1812 acct-port 1813
Device(config-if)#key cisco
```

Example: configure multi-domain

This example shows how to configure multi-domain MACsec:

```
Device>enable
Device#configure terminal
Device(config)#aaa new-model
!
Device(config)#aaa group server radius tests
Device(config)#server name RAD-1
!
Device(config)#aaa authentication dot1x default group tests
Device(config)#aaa authorization network default group tests
!
Device(config)#aaa server radius dynamic-author
Device(config)#client <ipv4 address> server-key cisco
!

Device(config)#dot1x system-auth-control
!

Device(config)#class-map type control subscriber match-all DOT1X

Device(config)#match method dot1x
!
Device(config)#class-map type control subscriber match-all DOT1X_FAILED

Device(config)#match method dot1x
 Device(config)#match result-type method dot1x authoritative
!
Device(config)#class-map type control subscriber match-all DOT1X_NO_RESP
 Device(config)#match method dot1x
 Device(config)#match result-type method dot1x agent-not-found
!
Device(config)#class-map type control subscriber match-all MAB
 Device(config)#match method mab
!
Device(config)#class-map type control subscriber match-all MAB_FAILED
 Device(config)#match method mab
 Device(config)#match result-type method mab authoritative
!
Device(config)#policy-map type control subscriber TEST4
Device(config)#event session-started match-all
Device(config)#10 class always do-until-failure
Device(config)#10 authenticate using dot1x priority 10
Device(config)#20 authenticate using mab priority 20
Device(config)#Device(config)event authentication-failure match-first
Device(config)#10 class DOT1X_FAILED do-until-failure
Device(config)#10 terminate dot1x
Device(config)#20 class MAB_FAILED do-until-failure
Device(config)#10 terminate mab
Device(config)#20 authenticate using dot1x priority 10
Device(config)#30 class DOT1X_NO_RESP do-until-failure
Device(config)#10 terminate dot1x
Device(config)#20 authentication-restart 60
Device(config)#40 class always do-until-failure
Device(config)#10 terminate mab
Device(config)#20 terminate dot1x
```

```
Device(config)#30 authentication-restart 60
Device(config)#event agent-found match-all
Device(config)#10 class always do-until-failure
Device(config)#10 terminate mab
Device(config)#20 authenticate using dot1x priority 10
Device(config)#event authentication-success match-all
Device(config)#10 class always do-until-failure
Device(config)#10 activate service-template DEFAULT_LINKSEC_POLICY_SHOULD_SECURE
!
Device(config-if)#interface GigabitEthernet1/2
Device(config-if)#macsec
Device(config-if)#switchport access vlan 613
Device(config-if)#switchport mode access
Device(config-if)#switchport voice vlan 612
Device(config-if)#access-session host-mode multi-domain
Device(config-if)#access-session port-control auto
Device(config-if)#mab
Device(config-if)#dot1x pae authenticator
Device(config-if)#spanning-tree portfast
Device(config-if)#service-policy type control subscriber TEST4
!
Device(config)#radius-server attribute 6 on-for-login-auth
Device(config)#radius-server attribute 8 include-in-access-req
Device(config)#radius-server attribute 25 access-request include
Device(config)#radius-server vsa send cisco-nas-port
!
Device(config)#radius server RAD-1
Device(config)#address ipv4 <ISE ipv4 address> auth-port 1812 acct-port 1813
Device(config)#key cisco
```

Additional References for MACsec Encryption

Standards and RFCs

Standard/RFC	Title
IEEE 802.1AE-2006	*Media Access Control (MAC) Security*
IEEE 802.1X-2010	*Port-Based Network Access Control*
IEEE 802.1AEbw-2013	*Media Access Control (MAC) Security (Amendment to IEEE 802.1AE-2006)—Extended Packet Numbering (XPN)*
IEEE 802.1Xbx-2014	*Port-Based Network Access Control (Amendment to IEEE 802.1X-2010)*
RFC 4493	*The AES-CMAC Algorithm*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

eEdge Integration with MACsec

The Media Access Control Security (MACsec) standard is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. The eEdge Integration with MACsec feature allows you to integrate the MACsec standard with enterprise edge (eEdge) devices to enhance Session Aware Networking capabilities. Session Aware Networking provides a policy and identity-based framework for edge devices to deliver flexible and scalable services to subscribers.

Prerequisites for eEdge Integration with MACsec

- Layer 2 encryption protocols like the IEEE 802.1AE Media Access Control Security (MACsec) standard must register with the eEdge session manager to receive disconnect notifications and perform cleanup.
- You must provision one virtual interface per secure association.

Restrictions for eEdge Integration with MACsec

- The Media Access Control Security (MACsec) standard is supported only in single-host and multihost modes. If a link layer security policy is configured as must-secure and the host mode is not configured as a single host or a multihost, the connection is closed.
- The MACsec standard is not supported in multi-authentication mode.
- The MACsec standard supports the 802.1AE encryption with MACsec Key Agreement (MKA) only on downlink ports for encryption between a MACsec-capable device and host devices.

Information About eEdge Integration with MACsec

The following sections provide information about eEdge Integration with MACsec feature.

Overview of MACsec

Media Access Control Security (MACsec) is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. Implementing the MACsec encryption standard enables support for the 802.1AE encryption with MACsec Key Agreement (MKA) on downlink ports for encryption between a MACsec-capable device and host devices. The MACsec-capable device also supports MACsec link layer device-to-device security by using Cisco TrustSec Network Device Admission Control (NDAC) and the Security Association Protocol (SAP) key exchange. Link layer security includes both packet authentication between devices and MACsec encryption between devices (encryption is optional).

MACsec Standard Encryption

The Media Access Control Security (MACsec) standard provides data link layer encryption over wired networks by using out-of-band methods for encryption keying. The MACsec Key Agreement (MKA) protocol provides the required session keys and manages the encryption keys. MKA and MACsec are implemented after a successful authentication by using the 802.1X Extensible Authentication Protocol (EAP) framework. Only host-facing links (links between network access devices and endpoint devices such as a PC or an IP phone) can be secured using MACsec.

A device that uses MACsec accepts either MACsec or non-MACsec frames, depending on the policy associated with the client. MACsec frames are encrypted and protected with an integrity check value (ICV). When the device receives frames from the client, it decrypts them and calculates the correct ICV by using session keys provided by MKA. The device compares the calculated value of the ICV to the ICV within the frame. If they are not identical, the frame is dropped. The device also encrypts and adds an ICV to any frame that is sent over a secured port (the access point used to provide the secure MAC service to a client) using the current session key.

The MKA protocol manages the encryption keys used by the underlying MACsec protocol. The basic requirements of MKA are defined in 802.1X-2010. The MKA protocol extends 802.1X to allow peer discovery with confirmation of mutual authentication and sharing of MACsec secret keys to protect data exchanged by peers.

EAP Implementation of MKA

The Extensible Authentication Protocol (EAP) framework implements MKA as a newly defined EAP-over-LAN (EAPOL) packet. EAP authentication produces a master session key (MSK) that is shared by both partners in the data exchange. Entering the EAP session ID generates a secure connectivity association key name (CKN). Because the device is the authenticator, it is also the key server, generating a random 128-bit secure association key (SAK), which it sends it to the client partner. The client is never a key server and can only interact with a single MKA entity, the key server. After key derivation and generation, the device sends periodic transports to the partner at a default interval of 2 seconds.

The packet body in an EAPOL Protocol Data Unit (PDU) is referred to as a MACsec Key Agreement PDU (MKPDU). MKA sessions and participants are deleted when the MKA lifetime (6 seconds) passes and no MKPDU is received from a participant. For example, if a client disconnects, the participant on the device continues to operate MKA until 6 seconds have elapsed after the last MKPDU is received from the client.

eEdge Integration with MACsec

The Media Access Control Security (MACsec) standard is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. The eEdge Integration with MACsec feature allows you to integrate the MACsec standard with enterprise edge (eEdge) devices to enhance Session Aware Networking capabilities. Session Aware Networking provides a policy and identity-based framework for edge devices to deliver flexible and scalable services to subscribers.

How to Configure eEdge Integration with MACsec

Integrating eEdge with MACsec

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service-template** *template-name* **Example:** `Device(config)# service-template dot1x-macsec-policy`	Defines a template that contains a set of service policy attributes to apply to subscriber sessions and enters service template configuration mode.
Step 4	**linksec policy** {**must-not-secure** | **must-secure** | **should-secure**} **Example:** `Device(config-service-template)# linksec policy must-secure`	Sets the link security policy as must-secure. • Must-secure policy authorizes the eEdge device port only if a secure MACsec session is established.
Step 5	**exit** **Example:** `Device(config-service-template)# exit`	Exits service template configuration mode and returns to global configuration mode.
Step 6	**policy-map type control subscriber** *control-policy-name* **Example:** `Device(config)# policy-map type control subscriber cisco-subscriber`	Defines a control policy for subscriber sessions and enters control policy-map event configuration mode.
Step 7	**event authentication-success** [**match-all** | **match-any**] **Example:** `Device(config-event-control-policymap)# event authentication-success match-all`	Specifies the type of event that triggers actions in a control policy if all authentication events are a match and enters control policy-map class configuration mode.
Step 8	*priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**] **Example:**	Specifies that the control class should execute the actions in a control policy, in the specified order, until one of the actions fails, and enters control policy-map action configuration mode.

	Command or Action	Purpose
	`Device(config-class-control-policymap)# 10 class always do-until-failure`	
Step 9	*action-number* **activate** {**policy type control subscriber** *control-policy-name* | **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** [**replace-all**]} **Example:** `Device(config-action-control-policymap)# 10 activate service-template dot1x-macsec-policy`	Activates a control policy on a subscriber session.
Step 10	**end** **Example:** `Device(config-action-control-policymap)# end`	Exits control policy-map action configuration mode and enters privileged EXEC mode.

Identifying Link Layer Security Failures

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**class-map type control subscriber** {**match-all** | **match-any** | **match-none**} *control-class-name* **Example:** `Device(config)# class-map type control subscriber match-all linksec-failed`	Creates a control class, which defines the conditions under which the actions of a control policy are executed and enters control class-map filter configuration mode.
Step 3	**match authorization-failure** {**domain-change-failed** | **linksec-failed**} **Example:** `Device(config-filter-control-classmap)# match authorization-failure linksec-failed`	Configures a match condition in a control class based on the type of authorization failure received from an authorization failed event of a link layer security failure.
Step 4	**exit** **Example:** `Device(config-class-control-policymap)# exit`	Exits control class-map filter configuration mode and enters global configuration mode.

	Command or Action	Purpose
Step 5	**policy-map type control subscriber** *control-policy-name* **Example:** `Device(config)# policy-map type control subscriber cisco-subscriber`	Defines a control policy for subscriber sessions and enters control policy-map event configuration mode.
Step 6	**event authentication-failure** [**match-all** | **match-any**] **Example:** `Device(config-event-control-policymap)# event authentication-failure match-all`	Specifies the type of event that triggers actions in a control policy if session authentication fails and enters control policy-map class configuration mode.
Step 7	*priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**] **Example:** `Device(config-class-control-policymap)# 10 class linksec-failed do-until-failure`	Specifies that the control class must execute the actions in a control policy, in the specified order, until one of the actions fails and enters control policy-map action configuration mode.
Step 8	**end** **Example:** `Device(config-action-control-policymap)# end`	Exits control policy-map action configuration mode and enters privileged EXEC mode.

Configuration Examples for eEdge Integration with MACsec

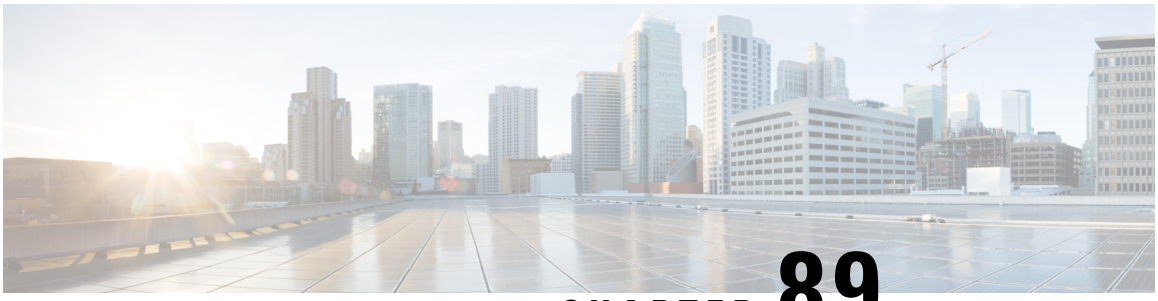
Example: Integrating eEdge with MACsec

```
Device> enable
Device# configure terminal
Device(config)# service-template dot1x-macsec-policy
Device(config-service-template)# linksec policy must-secure
Device(config-service-template)# exit
Device(config)# policy-map type control subscriber cisco-subscriber
Device(config-event-control-policymap)# event authentication-success match-all
Device(config-class-control-policymap)# 10 class always do-until-failure
Device(config-action-control-policymap)# 10 activate service-template dot1x-macsec-policy
Device(config-action-control-policymap)# end
```

Example: Identifying Linksec Failures

```
Device# configure terminal
Device(config)# class-map type control subscriber match-all linksec-failure
Device(config-filter-control-classmap)# match authorization-failure linksec-failed
Device(config-class-control-classmap)# exit
Device(config)# policy-map type control subscriber cisco-subscriber
Device(config-event-control-policymap)# event authentication-failure match-all
```

```
Device(config-class-control-policymap)# 10 class linksec-failed do-until-failure
Device(config-action-control-policymap)# end
```



CHAPTER 89

Secure Shell Version 2 Support

The Secure Shell Version 2 Support feature allows you to configure Secure Shell (SSH) Version 2. SSH runs on top of a reliable transport layer and provides strong authentication aind encryption capabilities. The only reliable transport that is defined for SSH is TCP. SSH provides a means to securely access and securely execute commands on another computer over a network. The Secure Copy Protocol (SCP) feature that is provided with SSH allows for the secure transfer of files.

Prerequisites for Secure Shell Version 2 Support

- Before configuring SSH, ensure that the required image is loaded on your device. The SSH server requires you to have a k9 (Triple Data Encryption Standard [3DES]) software image depending on your release.
- You have to use a SSH remote device that supports SSH Version 2 and connect to a Cisco device.
- SCP relies on authentication, authorization, and accounting (AAA) to function correctly. Therefore, AAA must be configured on the device to enable the secure copy protocol on the SSH Server.

Note The SSH Version 2 server and the SSH Version 2 client are supported on your Cisco software. (The SSH client is supported in k9 images.)

Restrictions for Secure Shell Version 2 Support

- Secure Shell (SSH) servers and SSH clients are supported in Triple Data Encryption Standard (3DES) software images.
- Execution Shell, remote command execution, and Secure Copy Protocol (SCP) are the only applications supported.

- Rivest, Shamir, and Adleman (RSA) key generation is an SSH server-side requirement. Devices that act as SSH clients need not generate RSA keys.
- The RSA key pair size must be greater than or equal to 768 bits.
- The following features are not supported:
  - Port forwarding
  - Compression

Information About Secure Shell Version 2 Support

Secure Shell Version 2

The Secure Shell Version 2 Support feature configures SSH Version 2 by default.

The **ip ssh version** command defines the SSH version to be configured. If you do not configure this command, SSH by default runs in compatibility mode; that is, both SSH Version 1 and SSH Version 2 connections are honored.

The **ip ssh rsa keypair-name** command enables an SSH connection using the Rivest, Shamir, and Adleman (RSA) keys that you have configured. Previously, SSH was linked to the first RSA keys that were generated (that is, SSH was enabled when the first RSA key pair was generated). This behavior still exists, but by using the **ip ssh rsa keypair-name** command, you can overcome this behavior. If you configure the **ip ssh rsa keypair-name** command with a key pair name, SSH is enabled if the key pair exists or SSH will be enabled if the key pair is generated later. If you use this command to enable SSH, you are not forced to configure a hostname and a domain name.

Note The login banner is supported in SSH Version 2.

Secure Shell Version 2 Enhancements

The SSH Version 2 Enhancements feature includes a number of additional capabilities such as supporting Virtual Routing and Forwarding (VRF)-Aware SSH, SSH debug enhancements, and Diffie-Hellman (DH) group exchange support.

Note The VRF-Aware SSH feature is supported depending on your release.

The Cisco SSH implementation has traditionally used 768-bit modulus, but with an increasing need for higher key sizes to accommodate DH Group 14 (2048 bits) and Group 16 (4096 bits) cryptographic applications, a message exchange between the client and the server to establish the favored DH group becomes necessary. The **ip ssh dh min size** command configures the modulus size on the SSH server. In addition to this, the **ssh** command was extended to add VRF awareness to the SSH client-side functionality through which the VRF

instance name in the client is provided with the IP address to look up the correct routing table and establish a connection.

Debugging was enhanced by modifying SSH debug commands. The **debug ip ssh** command was extended to simplify the debugging process. Before the simplification of the debugging process, this command printed all debug messages related to SSH regardless of what was specifically required. The behavior still exists, but if you configure the **debug ip ssh** command with a keyword, messages are limited to information specified by the keyword.

Secure Shell Version 2 Enhancements for RSA Keys

Cisco SSH Version 2 supports keyboard-interactive and password-based authentication methods. The SSH Version 2 Enhancements for RSA Keys feature also supports RSA-based public key authentication for the client and the server.

- User authentication: RSA-based user authentication uses a private/public key pair associated with each user for authentication. The user must generate a private/public key pair on the client and configure a public key on the Cisco SSH server to complete the authentication.

  An SSH user trying to establish credentials provides an encrypted signature using the private key. The signature and the user's public key are sent to the SSH server for authentication. The SSH server computes a hash over the public key provided by the user. The hash is used to determine if the server has a matching entry. If a match is found, an RSA-based message verification is performed using the public key. Hence, the user is authenticated or denied access based on the encrypted signature.

- Server authentication: While establishing an SSH session, the Cisco SSH client authenticates the SSH server by using the server host keys available during the key exchange phase. SSH server keys are used to identify the SSH server. These keys are created at the time of enabling SSH and must be configured on the client.

  For server authentication, the Cisco SSH client must assign a host key for each server. When the client tries to establish an SSH session with a server, the client receives the signature of the server as part of the key exchange message. If the strict host key checking flag is enabled on the client, the client checks if it has the host key entry corresponding to the server. If a match is found, the client tries to validate the signature by using the server host key. If the server is successfully authenticated, the session establishment continues; otherwise, it is terminated and displays a "Server Authentication Failed" message.

Note

- Storing public keys on a server uses memory; therefore, the number of public keys configurable on an SSH server is restricted to ten users, with a maximum of two public keys per user.
- RSA-based user authentication is supported by the Cisco server, but Cisco clients cannot propose public key as an authentication method. If the Cisco server receives a request from an open SSH client for RSA-based authentication, the server accepts the authentication request.
- For server authentication, configure the RSA public key of the server manually and configure the **ip ssh stricthostkeycheck** command on the Cisco SSH client.

SSH And Switch Access

Secure Shell (SSH) is a protocol that provides a secure, remote connection to a device. SSH provides more security for remote connections than Telnet does by providing strong encryption when a device is authenticated. This software release supports SSH Version 2 (SSHv2).

SSH functions the same in IPv6 as in IPv4. For IPv6, SSH supports IPv6 addresses and enables secure, encrypted connections with remote IPv6 nodes over an IPv6 transport.

SNMP Trap Generation

Depending on your release, Simple Network Management Protocol (SNMP) traps are generated automatically when an SSH session terminates if the traps have been enabled and SNMP debugging has been enabled.

Note When you configure the **snmp-server host** command, the IP address must be the address of the PC that has the SSH (telnet) client and that has IP connectivity to the SSH server.

You must also enable SNMP debugging using the **debug snmp packet** command to display the traps. The trap information includes information such as the number of bytes sent and the protocol that was used for the SSH session.

SSH Keyboard Interactive Authentication

The SSH Keyboard Interactive Authentication feature, also known as Generic Message Authentication for SSH, is a method that can be used to implement different types of authentication mechanisms. Basically, any currently supported authentication method that requires only user input can be performed with this feature. The feature is automatically enabled.

The following methods are supported:

- Password
- SecurID and hardware tokens printing a number or a string in response to a challenge sent by the server
- Pluggable Authentication Module (PAM)
- S/KEY (and other One-Time-Pads)

How to Configure Secure Shell Version 2 Support

Configuring a Device for SSH Version 2 Using a Hostname and Domain Name

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device>enable`	Enter your password, if prompted.
Step 2	**crypto key generate rsa** **Example:** `Device>crypto key generate rsa`	Enables the SSH server for local and remote authentication.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**hostname** *name* **Example:** `Device(config)# hostname IE35xx`	Configures a hostname for your device.
Step 5	**ip domain name** *name* **Example:** `Device(config)# ip domain name example.com`	Configures a domain name for your device.
Step 6	**ip ssh** [**time-out** *seconds* | **authentication-retries** *integer*] **Example:** `Device(config)# ip ssh time-out 120`	(Optional) Configures SSH control variables on your device.
Step 7	**ip ssh version** [**1** | **2**] **Example:** `Device(config)# ip ssh version 1`	(Optional) Specifies the version of SSH to be run on your device.
Step 8	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode. • Use **no hostname** command to return to the default host.

Configuring a Device for SSH Version 2 Using RSA Key Pairs

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
Step 2	**crypto key generate rsa usage-keys label** *key-label* **modulus** *modulus-size* **Example:** `Device>crypto key generate rsa usage-keys label sshkeys modulus 768`	Enables the SSH server for local and remote authentication on the device. • For SSH Version 2, the modulus size must be at least 768 bits. **Note** To delete the RSA key pair, use the **crypto key zeroize rsa** command. When you delete the RSA key pair, you automatically disable the SSH server.
Step 3	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 4	**ip ssh rsa keypair-name** *keypair-name* **Example:** `Device(config)#ip ssh rsa keypair-name sshkeys`	Specifies the RSA key pair to be used for SSH. **Note** A Cisco device can have many RSA key pairs.
Step 5	**ip ssh** [**time-out** *seconds* | **authentication-retries** *integer*] **Example:** `Device(config)#ip ssh time-out 12`	Configures SSH control variables on your device.
Step 6	**ip ssh version 2** **Example:** `Device(config)#ip ssh version 2`	Specifies the version of SSH to be run on the device.
Step 7	**exit** **Example:** `Device(config)#exit`	Exits global configuration mode and enters privileged EXEC mode.

Configuring the Cisco SSH Server to Perform RSA-Based User Authentication

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
Step 2	**crypto key generate rsa** **Example:** `Device>crypto key generate rsa`	Generates RSA key pairs.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**hostname** *name* **Example:** `Device(config)# hostname host1`	Specifies the hostname.
Step 5	**ip domain name** *name* **Example:** `host1(config)# ip domain name name1`	Defines a default domain name that the Cisco software uses to complete unqualified hostnames.
Step 6	**ip ssh pubkey-chain** **Example:** `host1(config)# ip ssh pubkey-chain`	Configures SSH-RSA keys for user and server authentication on the SSH server and enters public-key configuration mode. • The user authentication is successful if the RSA public key stored on the server is verified with the public or the private key pair stored on the client.
Step 7	**username** *username* **Example:** `host1(conf-ssh-pubkey)# username user1`	Configures the SSH username and enters public-key user configuration mode.
Step 8	**key-string** **Example:** `host1(conf-ssh-pubkey-user)# key-string`	Specifies the RSA public key of the remote peer and enters public-key data configuration mode. **Note** You can obtain the public key value from an open SSH client; that is, from the .ssh/id_rsa.pub file.
Step 9	**key-hash** *key-type key-name* **Example:** `host1(conf-ssh-pubkey-data)# key-hash ssh-rsa key1`	(Optional) Specifies the SSH key type and version. • The key type must be ssh-rsa for the configuration of private public key pairs. • This step is optional only if the **key-string** command is configured. • You must configure either the **key-string** command or the **key-hash** command.

	Command or Action	Purpose
		Note You can use a hashing software to compute the hash of the public key string, or you can also copy the hash value from another Cisco device. Entering the public key data using the **key-string** command is the preferred way to enter the public key data for the first time.
Step 10	**end** **Example:** `host1(conf-ssh-pubkey-data)# end`	Exits public-key data configuration mode and returns to privileged EXEC mode. • Use **no hostname** command to return to the default host.

Configuring the Cisco IOS SSH Client to Perform RSA-Based Server Authentication

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**crypto key generate rsa** **Example:** `Device>crypto key generate rsa`	Generates RSA key pairs.
Step 3	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 4	**hostname** *name* **Example:** `Device(config)#hostname host1`	Specifies the hostname.
Step 5	**ip domain name** *name* **Example:** `host1(config)#ip domain name name1`	Defines a default domain name that the Cisco software uses to complete unqualified hostnames.
Step 6	**ip ssh pubkey-chain** **Example:** `host1(config)#ip ssh pubkey-chain`	Configures SSH-RSA keys for user and server authentication on the SSH server and enters public-key configuration mode.

	Command or Action	Purpose
Step 7	**server** *server-name* **Example:** `host1(conf-ssh-pubkey)#server server1`	Enables the SSH server for public-key authentication on the device and enters public-key server configuration mode.
Step 8	**key-string** **Example:** `host1(conf-ssh-pubkey-server)#key-string`	Specifies the RSA public-key of the remote peer and enters public key data configuration mode. **Note** You can obtain the public key value from an open SSH client; that is, from the .ssh/id_rsa.pub file.
Step 9	**exit** **Example:** `host1(conf-ssh-pubkey-data)#exit`	Exits public-key data configuration mode and enters public-key server configuration mode.
Step 10	**key-hash** *key-type key-name* **Example:** `host1(conf-ssh-pubkey-server)#key-hash ssh-rsa key1`	(Optional) Specifies the SSH key type and version. • The key type must be ssh-rsa for the configuration of private/public key pairs. • This step is optional only if the **key-string** command is configured. • You must configure either the **key-string** command or the **key-hash** command. **Note** You can use a hashing software to compute the hash of the public key string, or you can copy the hash value from another Cisco device. Entering the public key data using the **key-string** command is the preferred way to enter the public key data for the first time.
Step 11	**end** **Example:** `host1(conf-ssh-pubkey-server)#end`	Exits public-key server configuration mode and returns to privileged EXEC mode.
Step 12	**configure terminal** **Example:** `host1#configure terminal`	Enters global configuration mode.
Step 13	**ip ssh stricthostkeycheck** **Example:** `host1(config)#ip ssh stricthostkeycheck`	Ensures that server authentication takes place. • The connection is terminated in case of a failure.

	Command or Action	Purpose
		• Use **no hostname** command to return to the default host.
Step 14	**end** **Example:** `host1(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Starting an Encrypted Session with a Remote Device

Note The device with which you want to connect must support a Secure Shell (SSH) server that has an encryption algorithm that is supported in Cisco software. Also, you need not enable your device. SSH can be run in disabled mode.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**ssh** [**-v 2** | **-c** {**aes128-ctr** | **aes192-ctr** | **aes256-ctr** | **aes128-cbc** | **3des** | **aes192-cbc** | **aes256-cbc**} | **-l** *user-id* | **-l** *user-id***:***vrf-name number ip-address ip-address* | **-l** *user-id***:***rotary number ip-address* | **-m** {**hmac-sha1-160** | **hmac-sha2-256** | **hmac-sha2-512** | **hmac-sha1-96**} | **-o numberofpasswordprompts** *n* | **-p** *port-num*] {*ip-addr* | *hostname*} [**command** | **-vrf**] **Example:** `Device# ssh -v 2 -c aes256-ctr -m hmac-sha1-160 -l user2 <ip address>`	Starts an encrypted session with a remote networking device.

Verifying the Status of the Secure Shell Connection

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password, if prompted.
Step 2	**show ssh** **Example:** `Device# show ssh`	Displays the status of SSH server connections.
Step 3	**exit** **Example:** `Device# exit`	Exits privileged EXEC mode and returns to user EXEC mode.

The following sample output from the **show ssh** command displays status of various Version 2 connections:

```
---------------------------------------------------------------------------
Device# show ssh

Connection Version Mode Encryption  Hmac                State
Username
1          2.0     IN   aes128-cbc  hmac-md5     Session started       lab
1          2.0     OUT  aes128-cbc  hmac-md5     Session started       lab.
---------------------------------------------------------------------------
```

Verifying the Secure Shell Version 2 Status

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**show ip ssh** **Example:** `Device# show ip ssh`	Displays the version and configuration data for SSH.
Step 3	**exit** **Example:** `Device# exit`	Exits privileged EXEC mode and returns to user EXEC mode.

Examples

The following sample output from the **show ip ssh** command displays the version of SSH that is enabled, the authentication timeout values, and the number of authentication retries for Version 2 connections:

```
---------------------------------------------------------------------------
Device# show ip ssh

SSH Enabled - version 2.0
Authentication timeout: 120 secs; Authentication retries: 3
---------------------------------------------------------------------------
```

Monitoring and Maintaining Secure Shell Version 2

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**debug ip ssh** **Example:** `Device# debug ip ssh`	Enables debugging of SSH.
Step 3	**debug snmp packet** **Example:** `Device# debug snmp packet`	Enables debugging of every SNMP packet sent or received by the device.

Example

The following sample output from the **debug ip ssh** command shows the connection is an SSH Version 2 connection:

```
Device# debug ip ssh

00:33:55: SSH1: starting SSH control process
00:33:55: SSH1: sent protocol version id SSH-1.99-Cisco-1.25
00:33:55: SSH1: protocol version id is - SSH-2.0-OpenSSH_2.5.2p2
00:33:55: SSH2 1: send: len 280 (includes padlen 4)
00:33:55: SSH2 1: SSH2_MSG_KEXINIT sent
00:33:55: SSH2 1: ssh_receive: 536 bytes received
00:33:55: SSH2 1: input: packet len 632
00:33:55: SSH2 1: partial packet 8, need 624, maclen 0
00:33:55: SSH2 1: ssh_receive: 96 bytes received
00:33:55: SSH2 1: partial packet 8, need 624, maclen 0
00:33:55: SSH2 1: input: padlen 11
00:33:55: SSH2 1: received packet type 20
00:33:55: SSH2 1: SSH2_MSG_KEXINIT received
```

```
00:33:55: SSH2: kex: client->server aes128-cbc hmac-md5 none
00:33:55: SSH2: kex: server->client aes128-cbc hmac-md5 none
00:33:55: SSH2 1: expecting SSH2_MSG_KEXDH_INIT
00:33:55: SSH2 1: ssh_receive: 144 bytes received
00:33:55: SSH2 1: input: packet len 144
00:33:55: SSH2 1: partial packet 8, need 136, maclen 0
00:33:55: SSH2 1: input: padlen 5
00:33:55: SSH2 1: received packet type 30
00:33:55: SSH2 1: SSH2_MSG_KEXDH_INIT received
00:33:55: SSH2 1: signature length 111
00:33:55: SSH2 1: send: len 384 (includes padlen 7)
00:33:55: SSH2: kex_derive_keys complete
00:33:55: SSH2 1: send: len 16 (includes padlen 10)
00:33:55: SSH2 1: newkeys: mode 1
00:33:55: SSH2 1: SSH2_MSG_NEWKEYS sent
00:33:55: SSH2 1: waiting for SSH2_MSG_NEWKEYS
00:33:55: SSH2 1: ssh_receive: 16 bytes received
00:33:55: SSH2 1: input: packet len 16
00:33:55: SSH2 1: partial packet 8, need 8, maclen 0
00:33:55: SSH2 1: input: padlen 10
00:33:55: SSH2 1: newkeys: mode 0
00:33:55: SSH2 1: received packet type 2100:33:55: SSH2 1: SSH2_MSG_NEWKEYS received
00:33:56: SSH2 1: ssh_receive: 48 bytes received
00:33:56: SSH2 1: input: packet len 32
00:33:56: SSH2 1: partial packet 16, need 16, maclen 16
00:33:56: SSH2 1: MAC #3 ok
00:33:56: SSH2 1: input: padlen 10
00:33:56: SSH2 1: received packet type 5
00:33:56: SSH2 1: send: len 32 (includes padlen 10)
00:33:56: SSH2 1: done calc MAC out #3
00:33:56: SSH2 1: ssh_receive: 64 bytes received
00:33:56: SSH2 1: input: packet len 48
00:33:56: SSH2 1: partial packet 16, need 32, maclen 16
00:33:56: SSH2 1: MAC #4 ok
00:33:56: SSH2 1: input: padlen 9
00:33:56: SSH2 1: received packet type 50
00:33:56: SSH2 1: send: len 32 (includes padlen 13)
00:33:56: SSH2 1: done calc MAC out #4
00:34:04: SSH2 1: ssh_receive: 160 bytes received
00:34:04: SSH2 1: input: packet len 64
00:34:04: SSH2 1: partial packet 16, need 48, maclen 16
00:34:04: SSH2 1: MAC #5 ok
00:34:04: SSH2 1: input: padlen 13
00:34:04: SSH2 1: received packet type 50
00:34:04: SSH2 1: send: len 16 (includes padlen 10)
00:34:04: SSH2 1: done calc MAC out #5
00:34:04: SSH2 1: authentication successful for lab
00:34:04: SSH2 1: input: packet len 64
00:34:04: SSH2 1: partial packet 16, need 48, maclen 16
00:34:04: SSH2 1: MAC #6 ok
00:34:04: SSH2 1: input: padlen 6
00:34:04: SSH2 1: received packet type 2
00:34:04: SSH2 1: ssh_receive: 64 bytes received
00:34:04: SSH2 1: input: packet len 48
00:34:04: SSH2 1: partial packet 16, need 32, maclen 16
00:34:04: SSH2 1: MAC #7 ok
00:34:04: SSH2 1: input: padlen 19
00:34:04: SSH2 1: received packet type 90
00:34:04: SSH2 1: channel open request
00:34:04: SSH2 1: send: len 32 (includes padlen 10)
00:34:04: SSH2 1: done calc MAC out #6
00:34:04: SSH2 1: ssh_receive: 192 bytes received
00:34:04: SSH2 1: input: packet len 64
00:34:04: SSH2 1: partial packet 16, need 48, maclen 16
```

```
00:34:04: SSH2 1: MAC #8 ok
00:34:04: SSH2 1: input: padlen 13
00:34:04: SSH2 1: received packet type 98
00:34:04: SSH2 1: pty-req request
00:34:04: SSH2 1: setting TTY - requested: height 24, width 80; set: height 24,
width 80
00:34:04: SSH2 1: input: packet len 96
00:34:04: SSH2 1: partial packet 16, need 80, maclen 16
00:34:04: SSH2 1: MAC #9 ok
00:34:04: SSH2 1: input: padlen 11
00:34:04: SSH2 1: received packet type 98
00:34:04: SSH2 1: x11-req request
00:34:04: SSH2 1: ssh_receive: 48 bytes received
00:34:04: SSH2 1: input: packet len 32
00:34:04: SSH2 1: partial packet 16, need 16, maclen 16
00:34:04: SSH2 1: MAC #10 ok
00:34:04: SSH2 1: input: padlen 12
00:34:04: SSH2 1: received packet type 98
00:34:04: SSH2 1: shell request
00:34:04: SSH2 1: shell message received
00:34:04: SSH2 1: starting shell for vty
00:34:04: SSH2 1: send: len 48 (includes padlen 18)
00:34:04: SSH2 1: done calc MAC out #7
00:34:07: SSH2 1: ssh_receive: 48 bytes received
00:34:07: SSH2 1: input: packet len 32
00:34:07: SSH2 1: partial packet 16, need 16, maclen 16
00:34:07: SSH2 1: MAC #11 ok
00:34:07: SSH2 1: input: padlen 17
00:34:07: SSH2 1: received packet type 94
00:34:07: SSH2 1: send: len 32 (includes padlen 17)
00:34:07: SSH2 1: done calc MAC out #8
00:34:07: SSH2 1: ssh_receive: 48 bytes received
00:34:07: SSH2 1: input: packet len 32
00:34:07: SSH2 1: partial packet 16, need 16, maclen 16
00:34:07: SSH2 1: MAC #12 ok
00:34:07: SSH2 1: input: padlen 17
00:34:07: SSH2 1: received packet type 94
00:34:07: SSH2 1: send: len 32 (includes padlen 17)
00:34:07: SSH2 1: done calc MAC out #9
00:34:07: SSH2 1: ssh_receive: 48 bytes received
00:34:07: SSH2 1: input: packet len 32
00:34:07: SSH2 1: partial packet 16, need 16, maclen 16
00:34:07: SSH2 1: MAC #13 ok
00:34:07: SSH2 1: input: padlen 17
00:34:07: SSH2 1: received packet type 94
00:34:07: SSH2 1: send: len 32 (includes padlen 17)
00:34:07: SSH2 1: done calc MAC out #10
00:34:08: SSH2 1: ssh_receive: 48 bytes received
00:34:08: SSH2 1: input: packet len 32
00:34:08: SSH2 1: partial packet 16, need 16, maclen 16
00:34:08: SSH2 1: MAC #14 ok
00:34:08: SSH2 1: input: padlen 17
00:34:08: SSH2 1: received packet type 94
00:34:08: SSH2 1: send: len 32 (includes padlen 17)
00:34:08: SSH2 1: done calc MAC out #11
00:34:08: SSH2 1: ssh_receive: 48 bytes received
00:34:08: SSH2 1: input: packet len 32
00:34:08: SSH2 1: partial packet 16, need 16, maclen 16
00:34:08: SSH2 1: MAC #15 ok
00:34:08: SSH2 1: input: padlen 17
00:34:08: SSH2 1: received packet type 94
00:34:08: SSH2 1: send: len 32 (includes padlen 16)
00:34:08: SSH2 1: done calc MAC out #12
00:34:08: SSH2 1: send: len 48 (includes padlen 18)
```

```
00:34:08: SSH2 1: done calc MAC out #13
00:34:08: SSH2 1: send: len 16 (includes padlen 6)
00:34:08: SSH2 1: done calc MAC out #14
00:34:08: SSH2 1: send: len 16 (includes padlen 6)
00:34:08: SSH2 1: done calc MAC out #15
00:34:08: SSH1: Session terminated normally
```

Configuration Examples for Secure Shell Version 2 Support

Example: Configuring Secure Shell Version 2

```
Device> enable
Device# configure terminal
Device(config)# ip ssh version 2
Device(config)# end
```

Example: Configuring Secure Shell Versions 1 and 2

```
Device> enable
Device# configure terminal
Device(config)# no ip ssh version
Device(config)# end
```

Example: Starting an Encrypted Session with a Remote Device

```
Device> enable
Device# ssh -v 2 -c aes256-cbc -m hmac-sha1-160 -l shaship <ipv4 address>
Device# exit
```

Example: Setting an SNMP Trap

The following example shows how to set an SNMP trap is set. The trap notification is generated automatically when the SSH session terminates. In the example, 10.1.1.1 is the IP address of the SSH client.

```
Device> enable
Device# configure terminal
Device(config)# snmp-server trap link switchover
Device(config)# snmp-server host 10.1.1.1 public tty
Device(config)# end
```

Examples: SSH Keyboard Interactive Authentication

Example: Enabling Client-Side Debugs

The following example shows that the client-side debugs are turned on, and the maximum number of prompts is six (three for the SSH keyboard interactive authentication method and three for the password authentication method).

```
Password:
Password:
Password:
```

```
Password:
Password:
Password: cisco123
Last login: Tue Dec 6 13:15:21 2005 from 10.76.248.213
user1@courier:~> exit
logout
[Connection to 10.76.248.200 closed by foreign host]
Device1# debug ip ssh client

SSH Client debugging is on

Device1# ssh -2 lab 10.1.1.3

Password:
*Nov 17 12:50:53.199: SSH0: sent protocol version id SSH-2-Cisco-1.25
*Nov 17 12:50:53.199: SSH CLIENT0: protocol version id is - SSH-2-Cisco-1.25
*Nov 17 12:50:53.199: SSH CLIENT0: sent protocol version id SSH-2-Cisco-1.25
*Nov 17 12:50:53.199: SSH CLIENT0: protocol version exchange successful
*Nov 17 12:50:53.203: SSH0: protocol version id is - SSH-2-Cisco-1.25
*Nov 17 12:50:53.335: SSH CLIENT0: key exchange successful and encryption on
*Nov 17 12:50:53.335: SSH2 CLIENT 0: using method keyboard-interactive
Password:
Password:
Password:
*Nov 17 12:51:01.887: SSH2 CLIENT 0: using method password authentication
Password:
Password: lab
Device2>

*Nov 17 12:51:11.407: SSH2 CLIENT 0: SSH2_MSG_USERAUTH_SUCCESS message received
*Nov 17 12:51:11.407: SSH CLIENT0: user authenticated
*Nov 17 12:51:11.407: SSH2 CLIENT 0: pty-req request sent
*Nov 17 12:51:11.411: SSH2 CLIENT 0: shell request sent
*Nov 17 12:51:11.411: SSH CLIENT0: session open
```

Example: Enabling ChPass with a Blank Password Change

In the following example, the ChPass feature is enabled, and a blank password change is accomplished using the SSH Keyboard Interactive Authentication method. A TACACS+ access control server (ACS) is used as the back-end AAA server.

```
Device> enable
Device1# ssh -l cisco 10.1.1.3

Password:
Old Password: cisco
New Password: cisco123
Re-enter New password: cisco123

Device2> exit

[Connection to 10.1.1.3 closed by foreign host]
```

Example: Enabling ChPass and Changing the Password on First Login

In the following example, the ChPass feature is enabled and TACACS+ ACS is used as the back-end server. The password is changed on the first login using the SSH keyboard interactive authentication method.

```
Device1> enable
Device1# ssh -2 cisco 10.1.1.3

Password: cisco
```

```
Your password has expired.
Enter a new one now.
New Password: cisco123
Re-enter New password: cisco123

Device2> exit

[Connection to 10.1.1.3 closed by foreign host]

Device1# ssh -2 cisco 10.1.1.3

Password:cisco1
Your password has expired.
Enter a new one now.
New Password: cisco
Re-enter New password: cisco12
The New and Re-entered passwords have to be the same.
Try again.
New Password: cisco
Re-enter New password: cisco

Device2>
```

Example: Enabling ChPass and Expiring the Password After Three Logins

In the following example, the ChPass feature is enabled and TACACS+ ACS is used as the back-end AAA server. The password expires after three logins using the SSH keyboard interactive authentication method.

```
Device# ssh -2 cisco. 10.1.1.3

Password: cisco

Device2> exit

[Connection to 10.1.1.3 closed by foreign host]

Device1# ssh -2 cisco 10.1.1.3

Password: cisco

Device2> exit

Device1# ssh -2 cisco 10.1.1.3

Password: cisco

Device2> exit

[Connection to 10.1.1.3 closed by foreign host]

Device1# ssh -2 cisco 10.1.1.3

Password: cisco
Your password has expired.
Enter a new one now.
New Password: cisco123
Re-enter New password: cisco123

Device2>
```

Example: SNMP Debugging

The following is sample output from the **debug snmp packet** command. The output provides SNMP trap information for an SSH session.

```
Device1# debug snmp packet

SNMP packet debugging is on
Device1# ssh -2 lab 10.0.0.2
Password:

Device2# exit

[Connection to 10.0.0.2 closed by foreign host]
Device1#
*Jul 18 10:18:42.619: SNMP: Queuing packet to 10.0.0.2
*Jul 18 10:18:42.619: SNMP: V1 Trap, ent cisco, addr 10.0.0.1, gentrap 6, spectrap 1
local.9.3.1.1.2.1 = 6
tcpConnEntry.1.10.0.0.1.22.10.0.0.2.55246 = 4
ltcpConnEntry.5.10.0.0.1.22.10.0.0.2.55246 = 1015
ltcpConnEntry.1.10.0.0.1.22.10.0.0.2.55246 = 1056
ltcpConnEntry.2.10.0.0.1.22.10.0.0.2.55246 = 1392
local.9.2.1.18.2 = lab
*Jul 18 10:18:42.879: SNMP: Packet sent via UDP to 10.0.0.2

Device1#
```

Examples: SSH Debugging Enhancements

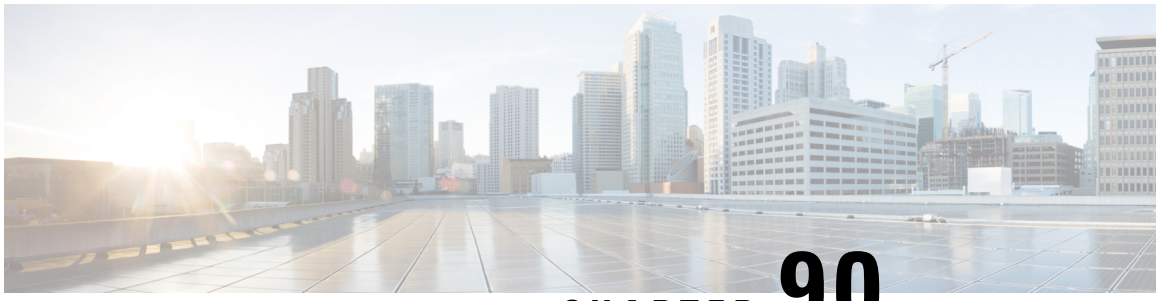
The following is sample output from the **debug ip ssh detail** command. The output provides debugging information about the SSH protocol and channel requests.

```
Device# debug ip ssh detail

00:04:22: SSH0: starting SSH control process
00:04:22: SSH0: sent protocol version id SSH-1.99-Cisco-1.25
00:04:22: SSH0: protocol version id is - SSH-1.99-Cisco-1.25
00:04:22: SSH2 0: SSH2_MSG_KEXINIT sent
00:04:22: SSH2 0: SSH2_MSG_KEXINIT received
00:04:22: SSH2:kex: client->server enc:aes128-cbc mac:hmac-sha1
00:04:22: SSH2:kex: server->client enc:aes128-cbc mac:hmac-sha1
00:04:22: SSH2 0: expecting SSH2_MSG_KEXDH_INIT
00:04:22: SSH2 0: SSH2_MSG_KEXDH_INIT received
00:04:22: SSH2: kex_derive_keys complete
00:04:22: SSH2 0: SSH2_MSG_NEWKEYS sent
00:04:22: SSH2 0: waiting for SSH2_MSG_NEWKEYS
00:04:22: SSH2 0: SSH2_MSG_NEWKEYS received
00:04:24: SSH2 0: authentication successful for lab
00:04:24: SSH2 0: channel open request
00:04:24: SSH2 0: pty-req request
00:04:24: SSH2 0: setting TTY - requested: height 24, width 80; set: height 24, width 80
00:04:24: SSH2 0: shell request
00:04:24: SSH2 0: shell message received
00:04:24: SSH2 0: starting shell for vty
00:04:38: SSH0: Session terminated normally
```

The following is sample output from the **debug ip ssh packet** command. The output provides debugging information about the SSH packet.

```
Device# debug ip ssh packet

00:05:43: SSH2 0: send:packet of  length 280 (length also includes padlen of 4)
```

```
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: input: total packet length of 280 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 24 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 4 bytes
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: input: total packet length of 144 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 16 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 6 bytes
00:05:43: SSH2 0: signature length 143
00:05:43: SSH2 0: send:packet of  length 448 (length also includes padlen of 7)
00:05:43: SSH2 0: send:packet of  length 16 (length also includes padlen of 10)
00:05:43: SSH2 0: newkeys: mode 1
00:05:43: SSH2 0: ssh_receive: 16 bytes received
00:05:43: SSH2 0: input: total packet length of 16 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 8 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 10 bytes
00:05:43: SSH2 0: newkeys: mode 0
00:05:43: SSH2 0: ssh_receive: 52 bytes received
00:05:43: SSH2 0: input: total packet length of 32 bytes
00:05:43: SSH2 0: partial packet length(block size)16 bytes,needed 16 bytes, maclen 20
00:05:43: SSH2 0: MAC compared for #3 :ok
```



CHAPTER 90

Configuring SSH File Transfer Protocol

Secure Shell (SSH) includes support for SSH File Transfer Protocol (SFTP), which is a new standard file transfer protocol introduced in SSHv2. This feature provides a secure and authenticated method for copying device configuration or device image files.

Prerequisites for SSH File Transfer Protocol

- SSH must be enabled.
- The **ip ssh source-interface** *interface-type interface-number* command must be configured.

Restrictions for SSH File Transfer Protocol

- The SFTP server is not supported.
- SFTP boot is not supported.
- The **sftp** option in the **install add** command is not supported.

Information About SSH Support over IPv6

SSH File Transfer Protocol Overview

The SFTP client functionality is provided as part of the SSH component and is always enabled on the corresponding device. Therefore, any SFTP server user with the appropriate permission can copy files to and from the device.

An SFTP client is VRF-aware; you can configure the secure FTP client to use the virtual routing and forwarding (VRF) associated with a particular source interface during connection attempts.

How to Configure SSH File Transfer Protocol

The following sections provide information about the various tasks that comprise an SFTP configuration.

Configuring SFTP

Perform the following steps:

Before you begin

To configure a Cisco device for SFTP client-side functionality, the **ip ssh source-interface** *interface-type interface-number* command must be configured first.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip ssh source-interface** *interface-type interface-number* **Example:** `Device(config)# ip ssh source-interface GigabitEthernet 1/1`	Defines the source IP for the SSH session.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	(Optional) Displays the SFTP client-side functionality.
Step 6	**debug ip sftp** **Example:** `Device# debug ip sftp`	(Optional) Enables SFTP debugging.

Configuring SFTP Username Password

To configure a username and password for SFTP, perform the following steps:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip sftp username***username* **Example:** `Device# ip sftp username cisco`	Defines the username.
Step 4	**ip sftp password***password* **Example:** `Device# ip sftp password 0 cisco`	Defines the password. Specify the encryption level. • 0 – Unencrypted password. • 0 – Encrypted password. • Line – Clear text password
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Performing an SFTP Copy Operation

SFTP copy takes the IP or hostname of the corresponding server if Domain Name System (DNS) is configured. To perform SFTP copy operations, use the following commands in privileged EXEC mode:

Command	Purpose
Device# **copy ios-file-system:file sftp://user:pwd@server-ip//filepath** Or Device# **copy ios-file-system: sftp:**	Copies a file from the local Cisco IOS file system to the server. Specify the username, password, IP address, and filepath of the server.

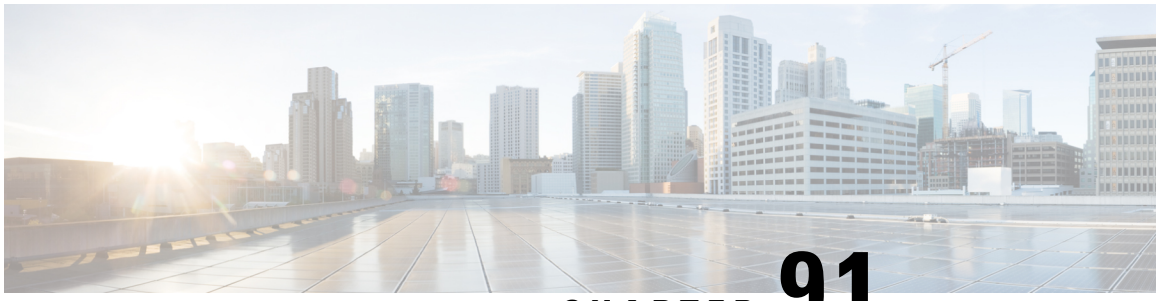
Command	Purpose
Device# **copy sftp://user:pwd@server-ip //filepath ios-file-system:file** Or Device# **copy sftp: ios-file-system:**	Copies the file from the server to the local Cisco IOS file system. Specify the username, password, IP address, and filepath of the server.

Configuration Examples for SSH Support over IPv6

Example: Configuring SSH File Transfer Protocol

The following example shows how to configure the client-side functionality of SFTP:

```
Device> enable
Device# configure terminal
Device(config)# ip ssh source-interface gigabitethernet 1/1
Device(config)# exit
```



CHAPTER 91

X.509v3 Certificates for SSH Authentication

The X.509v3 Certificates for SSH Authentication feature uses the X.509v3 digital certificates in server and user authentication at the secure shell (SSH) server side.

This module describes how to configure server and user certificate profiles for a digital certificate.

Prerequisites for X.509v3 Certificates for SSH Authentication

- The X.509v3 Certificates for SSH Authentication feature introduces the **ip ssh server algorithm authentication** command to replace the **ip ssh server authenticate user** command. If you use the **ip ssh server authenticate user** command, the following deprecation message is displayed.

  ```
  Warning: SSH command accepted but this CLI will be deprecated soon.
  Please move to new CLI "ip ssh server algorithm authentication".
  Please configure "default ip ssh server authenticate user" to make the CLI ineffective.
  ```

  Use the **default ip ssh server authenticate user** command to remove the **ip ssh server authenticate user** command from effect. The IOS secure shell (SSH) server then starts using the **ip ssh server algorithm authentication** command.

Restrictions for X.509v3 Certificates for SSH Authentication

- The X.509v3 Certificates for SSH Authentication feature implementation is applicable only on the Cisco IOS XE secure shell (SSH) server side.
- The SSH server supports only the x509v3-ssh-rsa algorithm-based certificate for server and user authentication.

Information About X.509v3 Certificates for SSH Authentication

The following section provides information about digital certificates, and server and user authentication.

Digital Certificates

The validity of the authentication depends upon the strength of the linkage between the public signing key and the identity of the signer. Digital certificates in the X.509v3 format (RFC5280) are used to provide identity management. A chain of signatures by a trusted root certification authority and its intermediate certificate authorities binds a given public signing key to a given digital identity.

Public key infrastructure (PKI) trustpoint helps manage the digital certificates. The association between the certificate and the trustpoint helps track the certificate. The trustpoint contains information about the certificate authority (CA), different identity parameters, and the digital certificate. Multiple trustpoints can be created to associate with different certificates.

Server and User Authentication using X.509v3

For server authentication, the Cisco IOS XE secure shell (SSH) server sends its own certificate to the SSH client for verification. This server certificate is associated with the trustpoint configured in the server certificate profile (ssh-server-cert-profile-server configuration mode).

For user authentication, the SSH client sends the user's certificate to the SSH server for verification. The SSH server validates the incoming user certificate using public key infrastructure (PKI) trustpoints configured in the server certificate profile (ssh-server-cert-profile-user configuration mode).

By default, certificate-based authentication is enabled for server and user at the SSH server end.

How to Configure X.509v3 Certificates for SSH Authentication

The following section provides information about how to configure X.509v3 Certificates for SSH Authentication.

Configuring the SSH Server to Use Digital Certificates for Server Authentication

To configure the SSH server to use digital certificates for server authentication, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ip ssh server algorithm hostkey** {**ecdsa-sha2-nistp256** | **ecdsa-sha2-nistp256** | **ecdsa-sha2-nistp384** | **ecdsa-sha2-nistp521** | **rsa-sha2-256** | **rsa-sha2-512** | **ssh-ed25519** | **ssh-rsa** | **x509v3-ecdsa-sha2-nistp256** | **x509v3-ecdsa-sha2-nistp384** | **x509v3-ecdsa-sha2-nistp521** | **x509v3-rsa2048-sha256** | **x509v3-ssh-rsa**} **Example:** `Device(config)# ip ssh server algorithm hostkey <enter-the-available-key>`	Defines the order of host key algorithms. Only the configured algorithm is negotiated with the secure shell (SSH) client. **Note** The IOS SSH server must have at least one configured host key algorithm: • ssh-rsa: public key based authentication • x509v3-ssh-rsa: certificate-based authentication
Step 4	**ip ssh server certificate profile** **Example:** `Device(config)# ip ssh server certificate profile`	Configures server certificate profile and user certificate profile and enters SSH certificate profile configuration mode.
Step 5	**server** **Example:** `Device(ssh-server-cert-profile)# server`	Configures server certificate profile and enters SSH server certificate profile server configuration mode.
Step 6	**trustpoint sign** *PKI-trustpoint-name* **Example:** `Device(ssh-server-cert-profile-server)# trustpoint sign trust1`	Attaches the public key infrastructure (PKI) trustpoint to the server certificate profile. The SSH server uses the certificate associated with this PKI trustpoint for server authentication.
Step 7	**ocsp-response include** **Example:** `Device(ssh-server-cert-profile-server)# ocsp-response include`	(Optional) Sends the Online Certificate Status Protocol (OCSP) response or OCSP stapling along with the server certificate. **Note** By default the **no** form of this command is configured and no OCSP response is sent along with the server certificate.
Step 8	**end** **Example:** `Device(ssh-server-cert-profile-server)# end`	Exits SSH server certificate profile server configuration mode and returns to privileged EXEC mode.

Configuring the SSH Server to Verify Digital Certificates for User Authentication

To configure the SSH Server to use digital certificates for user authentication, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip ssh server algorithm authentication** {**publickey** | **keyboard** | **password**} **Example:** `Device(config)# ip ssh server algorithm authentication publickey`	Defines the order of user authentication algorithms. Only the configured algorithm is negotiated with the secure shell (SSH) client. **Note** • The SSH server must have at least one configured user authentication algorithm. • To use the certificate method for user authentication, the **publickey** keyword must be configured. • The **ip ssh server algorithm authentication** command replaces the **ip ssh server authenticate user** command.
Step 4	**ip ssh server algorithm publickey** {**ecdsa-sha2-nistp256** | **ecdsa-sha2-nistp256** | **ecdsa-sha2-nistp384** | **ecdsa-sha2-nistp521** | **rsa-sha2-256** | **rsa-sha2-512** | **ssh-ed25519** | **ssh-rsa** | **x509v3-ecdsa-sha2-nistp256** | **x509v3-ecdsa-sha2-nistp384** | **x509v3-ecdsa-sha2-nistp521** | **x509v3-rsa2048-sha256** | **x509v3-ssh-rsa**} **Example:** `Device(config)# ip ssh server algorithm publickey <enter-the-available-key>`	Defines the order of public key algorithms. Only the configured algorithm is accepted by the SSH client for user authentication. **Note** The SSH client must have at least one configured public key algorithm: • ssh-rsa: public-key-based authentication • x509v3-ssh-rsa: certificate-based authentication
Step 5	**ip ssh server certificate profile** **Example:** `Device(config)# ip ssh server certificate profile`	Configures server certificate profile and user certificate profile and enters SSH certificate profile configuration mode.
Step 6	**user** **Example:** `Device(ssh-server-cert-profile)# user`	Configures user certificate profile and enters SSH server certificate profile user configuration mode.

	Command or Action	Purpose
Step 7	**trustpoint verify** *PKI-trustpoint-name* **Example:** `Device(ssh-server-cert-profile-user)# trustpoint verify trust2`	Configures the public key infrastructure (PKI) trustpoint that is used to verify the incoming user certificate. **Note** Configure multiple trustpoints by executing the same command multiple times. A maximum of 10 trustpoints can be configured.
Step 8	**ocsp-response required** **Example:** `Device(ssh-server-cert-profile-user)# ocsp-response required`	(Optional) Mandates the presence of the Online Certificate Status Protocol (OCSP) response with the incoming user certificate. **Note** By default the **no** form of this command is configured and the user certificate is accepted without an OCSP response.
Step 9	**end** **Example:** `Device(ssh-server-cert-profile-user)# end`	Exits SSH server certificate profile user configuration mode and returns to privileged EXEC mode.

Configuring Trustpoint Authentication and Creating Device Certificate

To configure trustpoint authentication and create device certificate, perform this procedure:

Note We recommend that you use a new RSA keypair name for the newly configured PKI certificate. If you want to reuse an existing RSA keypair name (that is associated with an old certificate) for a new PKI certificate, do either of the following:

- Do not regenerate a new RSA keypair with an existing RSA keypair name, reuse the existing RSA keypair name. Regenerating a new RSA keypair with an existing RSA keypair name will make all the certificates associated with the existing RSA keypair invalid.
- Manually remove the old PKI certificate configurations first, before reusing the existing RSA keypair name for the new PKI certificate.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Device(config)#crypto pki trustpoint trust1`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.
Step 4	**enrollment url** *url* **Example:** `Device(ca-trustpoint)#enrollment url http://10.1.1.10:80`	Specifies the URL of the CA on which your device should send certificate requests.
Step 5	**revocation-check none** **Example:** `Device(ca-trustpoint)#revocation-check none`	Specifies that certificate checking is ignored.
Step 6	**rsakeypair** *key-label* [*key-size* [*encryption-key-size*]] **Example:** `Device(ca-trustpoint)#rsakeypair trust1 2048`	(Optional) Specifies which key pair to associate with the certificate. A key pair with the *key-label* argument will be generated during enrollment if it does not already exist or if the **auto-enroll regenerate** command was issued. Specify the *key-size* argument for generating the key, and specify the *encryption-key-size* argument to request separate encryption, signature keys, and certificates. The *key-size* argument range is from 512 to 4096. The key-size and encryption-key-size must be the same size. Length of less than 2048 is not recommended. **Note** If this command is not enabled, the FQDN key pair is used.
Step 7	**exit** **Example:** `Device(ca-trustpoint)#exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 8	**crypto pki authenticate** *name* **Example:** `Device(config)#crypto pki authenticate trust1`	Retrieves the CA certificate and authenticates it. Check the certificate fingerprint if prompted. **Note**

	Command or Action	Purpose
		This command is optional if the CA certificate is already loaded into the configuration.
Step 9	**crypto pki enroll** *name* **Example:** `Device(config)#crypto pki enroll trust1`	Certificate request is sent to the certificate server and the server issues the ID or device certificate. You are prompted for enrollment information, such as whether to include the device FQDN and IP address in the certificate request.
Step 10	**show crypto pki certificates** **Example:** `Device# show crypto pki certificates verbose trust1`	(Optional) Displays information about your certificates, including any rollover certificates.

What to do next

For more information on how to install the certificate using other enrollment options, see Deploying RSA Keys Within a PKI.

Verifying Configuration for Server and User Authentication Using Digital Certificates

To verify configuration for server and user Authentication using digital certificates, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**show ip ssh** **Example:** `Device# show ip ssh` `SSH Enabled - version 1.99` `Authentication methods:publickey,keyboard-interactive,password` `Authentication Publickey Algorithms:x509v3-ssh-rsa,ssh-rsa` `Hostkey Algorithms:x509v3-ssh-rsa,ssh-rsa` `Authentication timeout: 120 secs;` `Authentication retries: 3` `Minimum expected Diffie Hellman key size : 1024 bits`	Displays the currently configured authentication methods. To confirm the use of certificate-based authentication, ensure that the x509v3-ssh-rsa algorithm is the configured host key algorithm.

Configuration Examples for X.509v3 Certificates for SSH Authentication

The following section provides examples for user and server authentication using digital certificates.

Example: Configuring the SSH Server to Use Digital Certificates for Server Authentication

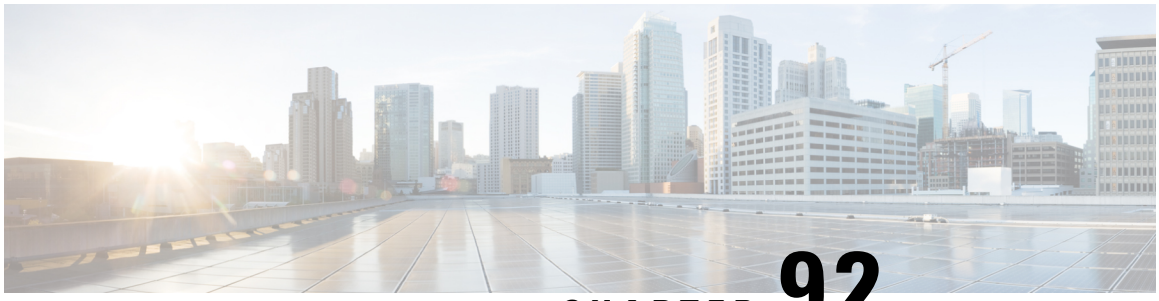
This example shows how to configure the SSH Server to use digital certificates for server authentication.

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm publickey ecdsa-sha2-nistp256
Device(config)# ip ssh server certificate profile
Device(ssh-server-cert-profile)# server
Device(ssh-server-cert-profile-server)# trustpoint sign trust1
Device(ssh-server-cert-profile-server)# end
```

Example: Configuring the SSH Server to Verify Digital Certificates for User Authentication

This example shows how to configure the SSH server to verify user's digital certificate for user authentication.

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm authentication publickey
Device(config)# ip ssh server algorithm publickey ecdsa-sha2-nistp256
Device(config)# ip ssh server certificate profile
Device(ssh-server-cert-profile)# user
Device(ssh-server-cert-profile-user)# trustpoint verify trust2
Device(ssh-server-cert-profile-user)# end
```



CHAPTER 92

SSH Algorithms for Common Criteria Certification

Restriction for SSH Algorithms for Common Criteria Certification

SHA1 is not supported.

Information About SSH Algorithms for Common Criteria Certification

This section provides information about the Secure Shell (SSH) Algorithms for Common Criteria Certification, the Cisco IOS SSH Server Algorithms and Cisco IOS SSH Client Algorithms.

SSH Algorithms for Common Criteria Certification

A Secure Shell (SSH) configuration enables a Cisco IOS SSH server and client to authorize the negotiation of only those algorithms that are configured from the allowed list, and the priority of the algorithms are based on the user configuration. If a remote party tries to negotiate using only those algorithms that are not part of the allowed list, the request is rejected and the session is not established.

Cisco IOS SSH Server Algorithms

Cisco IOS secure shell (SSH) servers support the encryption algorithms (Advanced Encryption Standard Counter Mode [AES-CTR], AES Cipher Block Chaining [AES-CBC], Triple Data Encryption Standard [3DES]), and Galois/Counter Mode (GCM) in the following order:

Supported Default Encryption Order:

1. chacha20-poly1305@openssh.com

2. aes128-gcm@openssh.com
3. aes256-gcm@openssh.com
4. aes128-gcm
5. aes256-gcm
6. aes128-ctr
7. aes192-ctr
8. aes256-ctr

Supported Non-Default Encryption:

- aes128-cbc
- aes192-cbc
- aes256-cbc
- 3des-cbc

Cisco IOS SSH servers support the Message Authentication Code (MAC) algorithms in the following order:

Supported Default HMAC Order:

1. hmac-sha2-256-etm@openssh.com
2. hmac-sha2-512-etm@openssh.com

Supported Non-Default HMAC:

- hmac-sha1
- hmac-sha2-256
- hmac-sha2-512

Cisco IOS SSH servers support the host key algorithms in the following order:

Supported Default Host Key Order:

1. rsa-sha2-512
2. rsa-sha2-256
3. ssh-rsa

Supported Non-Default Host Key:

- x509v3-ssh-rsa

Cisco IOS SSH servers support the Key Exchange (KEX) DH Group algorithms in the following default order:

Supported Default KEX DH Group Order:

1. curve25519-sha256

2. curve25519-sha256@libssh.org
3. ecdh-sha2-nistp256
4. ecdh-sha2-nistp384
5. ecdh-sha2-nistp521
6. diffie-hellman-group14-sha256
7. diffie-hellman-group16-sha512

Supported Non-Default KEX DH Group:

- diffie-hellman-group14-sha1

Cisco IOS SSH servers support the public key algorithms in the following default order:

Supported Default Public Key Order:

1. ssh-rsa
2. ecdsa-sha2-nistp256
3. ecdsa-sha2-nistp384
4. ecdsa-sha2-nistp521
5. ssh-ed25519
6. x509v3-ecdsa-sha2-nistp256
7. x509v3-ecdsa-sha2-nistp384
8. x509v3-ecdsa-sha2-nistp521
9. rsa-sha2-256
10. rsa-sha2-512
11. x509v3-rsa2048-sha256

Supported Non-Default Public Key:

- x509v3-ssh-rsa

Cisco IOS SSH Client Algorithms

Cisco IOS secure shell (SSH) clients support the the encryption algorithms (Advanced Encryption Standard counter mode [AES-CTR], AES Cipher Block Chaining [AES-CBC], Triple Data Encryption Standard [3DES]), and Galois/Counter Mode (GCM) in the following order:

Supported Default Encryption Order:

1. chacha20-poly1305@openssh.com
2. aes128-gcm@openssh.com
3. aes256-gcm@openssh.com

4. aes128-gcm
5. aes256-gcm
6. aes128-ctr
7. aes192-ctr
8. aes256-ctr

Supported Non-Default Encryption:

- aes128-cbc
- aes192-cbc
- aes256-cbc
- 3des-cbc

Cisco IOS SSH clients support the Message Authentication Code (MAC) algorithms in the following order:

Supported Default HMAC order:

1. hmac-sha2-256-etm@openssh.com
2. hmac-sha2-512-etm@openssh.com

Supported Non-Default HMAC:

- hmac-sha1
- hmac-sha2-256
- hmac-sha2-512

Cisco IOS SSH clients support the Key Exchange (KEX) DH Group algorithms in the following default order:

Supported Default KEX DH Group Order:

1. curve25519-sha256
2. curve25519-sha256@libssh.org
3. ecdh-sha2-nistp256
4. ecdh-sha2-nistp384
5. ecdh-sha2-nistp521
6. diffie-hellman-group14-sha256
7. diffie-hellman-group16-sha512

Supported Non-Default KEX DH Group:

- diffie-hellman-group14-sha1

How to Configure SSH Algorithms for Common Criteria Certification

This section provides information on how to configure and troubleshoot:

- Encryption key algorithm for a Cisco IOS SSH server and client
- MAC algorithm for a Cisco IOS SSH server and client
- Key Exchange DH Group algorithm for Cisco IOS SSH server and client
- Public Key algorithm for a Cisco IOS SSH server
- Host Key algorithm for a Cisco IOS SSH server

Configuring an Encryption Key Algorithm for a Cisco IOS SSH Server and Client

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip ssh** {**server** | **client**} **algorithm encryption** {**3des-cbc** | **aes128-cbc** | **aes128-ctr** | **aes128-gcm** | **aes128-gcm@openssh.com** | **aes192-cbc** | **aes192-ctr** | **aes256-cbc** | **aes256-ctr** | **aes256-gcm** | **aes256-gcm@openssh.com** | **chacha20-poly1305@openssh.com**} **Example:** `Device(config)# ip ssh server algorithm encryption 3des-cbc aes128-cbc aes128-ctr aes128-gcm aes128-gcm@openssh.com aes192-cbc aes192-ctr aes256-cbc aes256-ctr aes256-gcm aes256-gcm@openssh.com chacha20-poly1305@openssh.com` `Device(config)# ip ssh client algorithm encryption 3des-cbc aes128-cbc`	Defines the order of encryption algorithms in the SSH server and client. This order is presented during algorithm negotiation. **Note** • The Cisco IOS SSH server and client must have at least one configured encryption algorithm. • To disable one algorithm from the previously configured algorithm list, use the **no** form of this command. To disable more than one algorithm, use the **no** form of this command multiple times with different algorithm names. For a default configuration, use the default form of this command as shown below:

	Command or Action	Purpose
	aes128-ctr aes128-gcm aes128-gcm@openssh.com aes192-cbc aes192-ctr aes256-cbc aes256-ctr aes256-gcm aes256-gcm@openssh.com chacha20-poly1305@openssh.com	`Device(config)# ip ssh server algorithm encryption chacha20-poly1305@openssh.com aes128-gcm@openssh.com aes256-gcm@openssh.com aes128-gcm aes256-gcm aes128-ctr aes192-ctr aes256-ctr` `Device(config)# ip ssh client algorithm encryption chacha20-poly1305@openssh.com aes128-gcm@openssh.com aes256-gcm@openssh.com aes128-gcm aes256-gcm aes128-ctr aes192-ctr aes256-ctr`
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

If you try to disable the last encryption algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All encryption algorithms cannot be disabled
```

Configuring a MAC Algorithm for a Cisco IOS SSH Server and Client

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip ssh** {**server** | **client**} **algorithm mac** {**hmac-sha1** | **hmac-sha2-256** | **hmac-sha2-256-etm@openssh.com** | **hmac-sha2-512** | **hmac-sha2-512-etm@openssh.com**} **Example:** `Device(config)# ip ssh server algorithm mac hmac-sha2-256-etm hmac-sha2-512-etm hmac-sha2-256 hmac-sha2-512` `Device(config)# ip ssh client algorithm mac hmac-sha2-256-etm hmac-sha2-512-etm hmac-sha2-256 hmac-sha2-512`	Defines the order of MAC (Message Authentication Code) algorithms in the SSH server and client. This order is presented during algorithm negotiation. **Note** • The Cisco IOS SSH server and client must have at least one configured Hashed Message Authentication Code (HMAC) algorithm. • To disable one algorithm from the previously configured algorithm list, use

	Command or Action	Purpose
		the **no** form of this command. To disable more than one algorithm, use the **no** form of this command multiple times with different algorithm names. For default configuration, use the default form of this command as shown below: `Device(config)# ip ssh server algorithm mac hmac-sha2-256-etm@openssh.com hmac-sha2-512-etm@openssh.com` `Device(config)# ip ssh client algorithm mac hmac-sha2-256-etm@openssh.com hmac-sha2-512-etm@openssh.com`
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

If you try to disable the last MAC algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All mac algorithms cannot be disabled
```

Configuring a Key Exchange DH Group Algorithm for Cisco IOS SSH Server and Client

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip ssh** {**server** | **client**} **algorithm kex** {**curve25519-sha256** | **curve25519-sha256@libssh.org** | **diffie-hellman-group14-sha1** | **diffie-hellman-group14-sha256** | **diffie-hellman-group16-sha512** |	Defines the order of Key Exchange algorithms in the SSH server and client. This order is presented during algorithm negotiation. **Note** • The Cisco IOS SSH server and client must have at least one configured KEX algorithm.

	Command or Action	Purpose
	ecdh-sha2-nistp256 | **ecdh-sha2-nistp384**|**ecdh-sha2-nistp521** } **Example:** `Device(config)# ip ssh server algorithm kex curve25519-sha256@libssh.org diffie-hellman-group14-sha1 ecdh-sha2-nistp256 ecdh-sha2-nistp384 ecdh-sha2-nistp521` `Device(config)# ip ssh client algorithm kex curve25519-sha256@libssh.org diffie-hellman-group14-sha1 ecdh-sha2-nistp256 ecdh-sha2-nistp384 ecdh-sha2-nistp521`	• To disable one algorithm from the previously configured algorithm list, use the **no** form of this command. To disable more than one algorithm, use the **no** form of this command multiple times with different algorithm names. For default configuration, use the default form of this command as shown below: `Device(config)# ip ssh server algorithm kex curve25519-sha256 curve25519-sha256@libssh.org ecdh-sha2-nistp256 ecdh-sha2-nistp384 ecdh-sha2-nistp521 diffie-hellman-group14-sha256 diffie-hellman-group16-sha512` `Device(config)# ip ssh client algorithm kex curve25519-sha256 curve25519-sha256@libssh.org ecdh-sha2-nistp256 ecdh-sha2-nistp384 ecdh-sha2-nistp521 diffie-hellman-group14-sha256 diffie-hellman-group16-sha512`
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

If you try to disable the last KEX algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All KEX algorithms cannot be disabled
```

Configuring a Public Key Algorithm for a Cisco IOS SSH Server

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip ssh server algorithm publickey** {**ecdsa-sha2-nistp256** | **ecdsa-sha2-nistp384** | **ecdsa-sha2-nistp521** | **rsa-sha2-256** | **rsa-sha2-512** | **ssh-ed25519** | **ssh-rsa** | **x509v3-ecdsa-sha2-nistp256** | **x509v3-ecdsa-sha2-nistp384** | **x509v3-ecdsa-sha2-nistp521** | **x509v3-rsa2048-sha256** | **x509v3-ssh-rsa**} **Example:** `Device(config)# ip ssh server algorithm publickey ecdsa-sha2-nistp256 ecdsa-sha2-nistp384 ecdsa-sha2-nistp521 rsa-sha2-256 rsa-sha2-512 ssh-ed25519 ssh-rsa x509v3-ecdsa-sha2-nistp256 x509v3-ecdsa-sha2-nistp384 x509v3-ecdsa-sha2-nistp521 x509v3-rsa2048-sha256 x509v3-ssh-rsa`	Defines the order of public key algorithms in the SSH server. This order is presented during algorithm negotiation. **Note** • The Cisco IOS SSH server must have at least one configured public key algorithm. • To disable one algorithm from the previously configured algorithm list, use the **no** form of this command. To disable more than one algorithm, use the **no** form of this command multiple times with different algorithm names. For default configuration, use the default form of this command as shown below: `Device(config)# ip ssh server algorithm publickey ssh-rsa ecdsa-sha2-nistp256 ecdsa-sha2-nistp384 ecdsa-sha2-nistp521 ssh-ed25519 x509v3-ecdsa-sha2-nistp256 x509v3-ecdsa-sha2-nistp384 x509v3-ecdsa-sha2-nistp521 rsa-sha2-256 rsa-sha2-512 x509v3-rsa2048-sha256`
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

If you try to disable the last public key algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All public key algorithms cannot be disabled
```

Configuring a Host Key Algorithm for a Cisco IOS SSH Server

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip ssh server algorithm hostkey** {**rsa-sha2-512** | **rsa-sha2-256** | **ssh-rsa** | **x509v3-ssh-rsa**} **Example:** `Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa rsa-sha2-512 rsa-sha2-256 ssh-rsa`	Defines the order of host key algorithms. Only the configured algorithm is negotiated with the Cisco IOS secure shell (SSH) server. **Note** • The Cisco IOS SSH server must have at least one configured host key algorithm. • To disable one algorithm from the previously configured algorithm list, use the **no** form of this command. To disable more than one algorithm, use the **no** form of this command multiple times with different algorithm names. For default configuration, use the default form of this command as shown below: `Device(config)# ip ssh server algorithm hostkey rsa-sha2-512 rsa-sha2-256 ssh-rsa`
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

If you try to disable the last host key algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All hostkey algorithms cannot be disabled
```

Configuration Examples For SSH Algorithms for Common Criteria Certification

This section provides configuration examples for SSH algorithms for common certification.

Example: Configuring Encryption Key Algorithms for a Cisco IOS SSH Server

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm encryption 3des-cbc aes128-cbc aes128-ctr aes128-gcm
 aes128-gcm@openssh.com aes192-cbc aes192-ctr aes256-cbc aes256-ctr aes256-
gcm aes256-gcm@openssh.com chacha20-poly1305@openssh.com
Device(config)# end
```

Example: Configuring Encryption Key Algorithms for a Cisco IOS SSH Client

```
Device> enable
Device# configure terminal
Device(config)# ip ssh client algorithm encryption 3des-cbc aes128-cbc aes128-ctr aes128-gcm
 aes128-gcm@openssh.com aes192-cbc aes192-ctr aes256-cbc aes256-ctr aes256-
gcm aes256-gcm@openssh.com chacha20-poly1305@openssh.com
Device(config)# end
```

Example: Configuring MAC Algorithms for a Cisco IOS SSH Server

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm mac hmac-sha1 hmac-sha2-256
hmac-sha2-256-etm@openssh.com hmac-sha2-512-etm hmac-sha2-512-etm@openssh.com
Device(config)# end
```

Example: Configuring Key Exchange DH Group for a Cisco IOS SSH Server

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm kex ecdh-sha2-nistp256 ecdh-sha2-nistp384
ecdh-sha2-nistp521 diffie-hellman-group14-sha1 curve25519-sha256@libssh.org
Device(config)# end
```

Example: Configuring Encryption Public Key Algorithms for a Cisco IOS SSH Server

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm publickey ecdsa-sha2-nistp256 ecdsa-sha2-nistp384
ecdsa-sha2-nistp521 rsa-sha2-256 rsa-sha2-512 ssh-ed25519 ssh-rsa x509v3-ecdsa-sha2-nistp256
 x509v3-ecdsa-sha2-nistp384 x509v3-ecdsa-sha2-nistp521 x509v3-rsa2048-sha256 x509v3-ssh-rsa
Device(config)# end
```

The following example shows how to return to the default behavior in which all public key algorithms are enabled in the predefined order:

```
Device> enable
Device# configure terminal
Device(config)# default ip ssh server algorithm publickey
Device(config)# end
```

Example: Configuring Host Key Algorithms for a Cisco IOS SSH Server

```
Device> enable
Device# configure terminal
```

```
Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa rsa-sha2-512 rsa-sha2-256
ssh-rsaa
Device(config)# end
```

Verifying SSH Algorithms for Common Criteria Certification

Procedure

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **show ip ssh**

Displays configured Secure Shell (SSH) encryption, host key, and Message Authentication Code (MAC) algorithms.

Example:

The following sample output from the **show ip ssh** command shows the encryption algorithms configured in the default order:

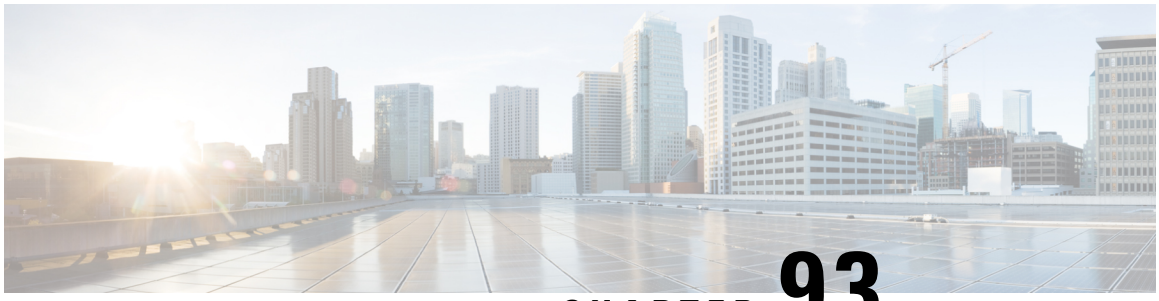
```
Device# show ip ssh

Encryption Algorithms: aes128-ctr aes192-ctr aes256-ctr aes128-cbc aes192-cbc aes256-cbc
3des
```

The following sample output from the **show ip ssh** command shows the MAC algorithms configured in the default order:

```
Device# show ip ssh

MAC Algorithms: hmac-sha2-256, hmac-sha2-512
```

The following sample output from the **show ip ssh** command shows the host key algorithms configured in the default order:

```
Device# show ip ssh

Hostkey Algorithms: x509v3-ssh-rsa, ssh-rsa
```



CHAPTER 93

Configuring Secure Socket Layer HTTP

Information About Secure Socket Layer HTTP

Secure HTTP Servers and Clients Overview

On a secure HTTP connection, data to and from an HTTP server is encrypted before being sent over the Internet. HTTP with SSL encryption provides a secure connection to allow such functions as configuring a switch from a Web browser. Cisco's implementation of the secure HTTP server and secure HTTP client uses an implementation of SSL Version 3.0 with application-layer encryption. HTTP over SSL is abbreviated as HTTPS; the URL of a secure connection begins with https:// instead of http://.

The primary role of the HTTP secure server (the switch) is to listen for HTTPS requests on a designated port (the default HTTPS port is 443) and pass the request to the HTTP 1.1 Web server. The HTTP 1.1 server processes requests and passes responses (pages) back to the HTTP secure server, which, in turn, responds to the original request.

The primary role of the HTTP secure client (the web browser) is to respond to Cisco IOS application requests for HTTPS User Agent services, perform HTTPS User Agent services for the application, and pass the response back to the application.

Certificate Authority Trustpoints

Certificate authorities (CAs) manage certificate requests and issue certificates to participating network devices. These services provide centralized security key and certificate management for the participating devices. Specific CA servers are referred to as *trustpoints*.

When a connection attempt is made, the HTTPS server provides a secure connection by issuing a certified X.509v3 certificate, obtained from a specified CA trustpoint, to the client. The client (usually a Web browser), in turn, has a public key that allows it to authenticate the certificate.

For secure HTTP connections, we highly recommend that you configure a CA trustpoint. If a CA trustpoint is not configured for the device running the HTTPS server, the server certifies itself and generates the needed RSA key pair. Because a self-certified (self-signed) certificate does not provide adequate security, the connecting

client generates a notification that the certificate is self-certified, and the user has the opportunity to accept or reject the connection. This option is useful for internal network topologies (such as testing).

If you do not configure a CA trustpoint, when you enable a secure HTTP connection, either a temporary or a persistent self-signed certificate for the secure HTTP server (or client) is automatically generated.

- If the switch is not configured with a hostname and a domain name, a temporary self-signed certificate is generated. If the switch reboots, any temporary self-signed certificate is lost, and a new temporary new self-signed certificate is assigned.
- If the switch has been configured with a host and domain name, a persistent self-signed certificate is generated. This certificate remains active if you reboot the switch or if you disable the secure HTTP server so that it will be there the next time you re-enable a secure HTTP connection.

Note The certificate authorities and trustpoints must be configured on each device individually. Copying them from other devices makes them invalid on the switch.

When a new certificate is enrolled, the new configuration change is not applied to the HTTPS server until the server is restarted. You can restart the server using the **reload** command. On restarting the server, the switch starts using the new certificate.

If a self-signed certificate has been generated, this information is included in the output of the **show running-config** privileged EXEC command. This is a partial sample output from that command displaying a self-signed certificate.

```
Device# show running-config

Building configuration...

<output truncated>

crypto pki trustpoint TP-self-signed-3080755072
 enrollment selfsigned
 subject-name cn=IOS-Self-Signed-Certificate-3080755072
 revocation-check none
 rsakeypair TP-self-signed-3080755072
!
!
crypto ca certificate chain TP-self-signed-3080755072
 certificate self-signed 01
  3082029F 30820208 A0030201 02020101 300D0609 2A864886 F70D0101 04050030
  59312F30 2D060355 04031326 494F532D 53656C66 2D536967 6E65642D 43657274
  69666963 6174652D 33303830 37353530 37323126 30240609 2A864886 F70D0109
  02161743 45322D33 3535302D 31332E73 756D6D30 342D3335 3530301E 170D3933
  30333031 30303030 35395A17 0D323030 31303130 30303030 305A3059 312F302D

<output truncated>
```

You can remove this self-signed certificate by disabling the secure HTTP server and entering the **no crypto pki trustpoint TP-self-signed-30890755072** global configuration command. If you later re-enable a secure HTTP server, a new self-signed certificate is generated.

Note The values that follow *TP self-signed* depend on the serial number of the device.

You can use the **ip http secure-client-auth** command (optional) to allow the HTTPS server to request an X.509v3 certificate from the client. Authenticating the client provides more security than server authentication by itself.

Note CA's self-signed root certificate must be configured for successful authentication of the client.

CipherSuites

A CipherSuite specifies the encryption algorithm and the digest algorithm to use on a SSL connection. When connecting to the HTTPS server, the client Web browser offers a list of supported CipherSuites, and the client and server negotiate the best encryption algorithm to use from those on the list that are supported by both. For example, Netscape Communicator 4.76 supports U.S. security with RSA Public Key Cryptography, MD2, MD5, RC2-CBC, RC4, DES-CBC, and DES-EDE3-CBC.

For the best possible encryption, you should use a client browser that supports 128-bit encryption, such as Microsoft Internet Explorer Version 5.5 (or later) or Netscape Communicator Version 4.76 (or later). The SSL_RSA_WITH_DES_CBC_SHA CipherSuite provides less security than the other CipherSuites, as it does not offer 128-bit encryption.

The more secure and more complex CipherSuites require slightly more processing time. This list defines the CipherSuites supported by the switch and ranks them from fastest to slowest in terms of router processing load (speed):

1. SSL_RSA_WITH_DES_CBC_SHA—RSA key exchange (RSA Public Key Cryptography) with DES-CBC for message encryption and SHA for message digest
2. SSL_RSA_WITH_NULL_SHA key exchange with NULL for message encryption and SHA for message digest (only for SSL 3.0).
3. SSL_RSA_WITH_NULL_MD5 key exchange with NULL for message encryption and MD5 for message digest (only for SSL 3.0).
4. SSL_RSA_WITH_RC4_128_MD5—RSA key exchange with RC4 128-bit encryption and MD5 for message digest
5. SSL_RSA_WITH_RC4_128_SHA—RSA key exchange with RC4 128-bit encryption and SHA for message digest
6. SSL_RSA_WITH_3DES_EDE_CBC_SHA—RSA key exchange with 3DES and DES-EDE3-CBC for message encryption and SHA for message digest
7. SSL_RSA_WITH_AES_128_CBC_SHA—RSA key exchange with AES 128-bit encryption and SHA for message digest (only for SSL 3.0).
8. SSL_RSA_WITH_AES_256_CBC_SHA—RSA key exchange with AES 256-bit encryption and SHA for message digest (only for SSL 3.0).
9. SSL_RSA_WITH_DHE_AES_128_CBC_SHA—RSA key exchange with AES 128-bit encryption and SHA for message digest (only for SSL 3.0).
10. SSL_RSA_WITH_DHE_AES_256_CBC_SHA—RSA key exchange with AES 256-bit encryption and SHA for message digest (only for SSL 3.0).

Note The latest versions of Chrome do not support the four original cipher suites, thus disallowing access to both web GUI and guest portals.

RSA (in conjunction with the specified encryption and digest algorithm combinations) is used for both key generation and authentication on SSL connections. This usage is independent of whether or not a CA trustpoint is configured.

Default SSL Configuration

The following guidelines apply to the default SSL configuration:

- The standard HTTP server is enabled.
- SSL is enabled.
- No CA trustpoints are configured.
- No self-signed certificates are generated.

SSL Configuration Guidelines

When SSL is used in a switch cluster, the SSL session terminates at the cluster commander. Cluster member switches must run standard HTTP.

Before you configure a CA trustpoint, you should ensure that the system clock is set. If the clock is not set, the certificate is rejected due to an incorrect date.

How to Configure Secure Socket Layer HTTP

Configuring a CA Trustpoint

For secure HTTP connections, we recommend that you configure an official CA trustpoint. A CA trustpoint is more secure than a self-signed certificate.

Beginning in privileged EXEC mode, follow these steps to configure a CA Trustpoint:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**crypto key generate rsa** **Example:**	(Optional) Generates an RSA key pair. RSA key pairs are required before you can obtain a certificate for the switch. RSA key pairs are

	Command or Action	Purpose
	`Device>crypto key generate rsa`	generated automatically. You can use this command to regenerate the keys, if needed.
Step 3	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 4	**hostname** *hostname* **Example:** `Device(config)#hostname your_hostname`	Specifies the hostname of the switch (required only if you have not previously configured a hostname). The hostname is required for security keys and certificates.
Step 5	**ip domain name** *domain-name* **Example:** `Device(config)#ip domain name your_domain`	Specifies the IP domain name of the switch (required only if you have not previously configured an IP domain name). The domain name is required for security keys and certificates.
Step 6	**crypto ca trustpoint** *name* **Example:** `Device(config)#crypto ca trustpoint your_trustpoint`	Specifies a local configuration name for the CA trustpoint and enter CA trustpoint configuration mode.
Step 7	**enrollment url** *url* **Example:** `Device(ca-trustpoint)#enrollment url http://your_server:80`	Specifies the URL to which the switch should send certificate requests.
Step 8	**enrollment http-proxy** *host-name port-number* **Example:** `Device(ca-trustpoint)#enrollment http-proxy your_host 49`	(Optional) Configures the switch to obtain certificates from the CA through an HTTP proxy server. • For *host-name* , specify the proxy server used to get the CA. • For *port-number*, specify the port number used to access the CA.
Step 9	**crl query** *url* **Example:** `Device(ca-trustpoint)#crl query ldap://your_host:49`	Configures the switch to request a certificate revocation list (CRL) to ensure that the certificate of the peer has not been revoked.
Step 10	**primary** *name* **Example:** `Device(ca-trustpoint)#primary your_trustpoint`	(Optional) Specifies that the trustpoint should be used as the primary (default) trustpoint for CA requests. • For *name*, specify the trustpoint that you just configured.

	Command or Action	Purpose
Step 11	**exit** **Example:** `Device(ca-trustpoint)#exit`	Exits CA trustpoint configuration mode and return to global configuration mode.
Step 12	**crypto ca authentication** *name* **Example:** `Device(config)#crypto ca authentication your_trustpoint`	Authenticates the CA by getting the public key of the CA. Use the same name used in Step 5.
Step 13	**crypto ca enroll** *name* **Example:** `Device(config)#crypto ca enroll your_trustpoint`	Obtains the certificate from the specified CA trustpoint. This command requests a signed certificate for each RSA key pair.
Step 14	**end** **Example:** `Device(config)#end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring the Secure HTTP Server

Beginning in privileged EXEC mode, follow these steps to configure a secure HTTP server:

Before you begin

If you are using a certificate authority for certification, you should use the previous procedure to configure the CA trustpoint on the switch before enabling the HTTP server. If you have not configured a CA trustpoint, a self-signed certificate is generated the first time that you enable the secure HTTP server. After you have configured the server, you can configure options (path, access list to apply, maximum number of connections, or timeout policy) that apply to both standard and secure HTTP servers.

To verify the secure HTTP connection by using a Web browser, enter https://*URL*, where the URL is the IP address or hostname of the server switch. If you configure a port other than the default port, you must also specify the port number after the URL. For example:

Note AES256_SHA2 is not supported.

```
https://209.165.129:1026
```

or

```
https://host.domain.com:1026
```

The existing **ip http access-class** *access-list-number* command for specifying the access-list(Only IPv4 ACLs) is going to be deprecated. You can still use this command to specify an access list to allow access to the HTTP server. Two new commands have been introduced to enable support for specifying IPv4 and IPv6 ACLs. These are **ip http access-class ipv4** *access-list-name* | *access-list-number* for specifying IPv4 ACLs and

ip http access-class ipv6 *access-list-name* for specifying IPv6 ACLs. We recommend using the new CLI to avoid receiving warning messages.

Note the following considerations for specifying access-lists:

- If you specify an access-list that does not exist, the configuration takes place but you receive the below warning message:

  ```
  ACL being attached does not exist, please configure it
  ```

- If you use **ip http access-class ipv4** *access-list-name* | *access-list-number* or **ip http access-class ipv6** *access-list-name* , and an access-list was already configured using **ip http access-class** , the below warning message appears:

  ```
  Removing ip http access-class <access-list-number>
  ```

ip http access-class *access-list-number* and **ip http access-class ipv4** *access-list-name* | *access-list-number* share the same functionality. Each command overrides the configuration of the previous command. The following combinations between the configuration of the two commands explain the effect on the running configuration:

- If **ip http access-class** *access-list-number* is already configured and you try to configure using **ip http access-class ipv4** *access-list-number* command, the configuration of **ip http access-class** *access-list-number* will be removed and the configuration of **ip http access-class ipv4** *access-list-number* will be added to the running configuration.
- If **ip http access-class** *access-list-number* is already configured and you try to configure using **ip http access-class ipv4** *access-list-name* command, the configuration of **ip http access-class** *access-list-number* will be removed and the configuration of **ip http access-class ipv4** *access-list-name* will be added to the running configuration.
- If **ip http access-class ipv4** *access-list-number* is already configured and you try to configure using **ip http access-class** *access-list-name*, the configuration of **ip http access-class ipv4** *access-list-number* will be removed from configuration and the configuration of **ip http access-class** *access-list-name* will be added to the running configuration.
- If **ip http access-class ipv4** *access-list-name* is already configured and you try to configure using **ip http access-class** *access-list-number*, the configuration of **ip http access-class ipv4** *access-list-name* will be removed from the configuration and the configuration of **ip http access-class** *access-list-number* will be added to the running configuration.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**show ip http server status** **Example:** `Device# show ip http server status`	(Optional) Displays the status of the HTTP server to determine if the secure HTTP server feature is supported in the software. You should see one of these lines in the output: `HTTP secure server capability: Present`

	Command or Action	Purpose
		or `HTTP secure server capability: Not present`
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**ip http secure-server** **Example:** `Device(config)# ip http secure-server`	Enables the HTTPS server if it has been disabled. The HTTPS server is enabled by default.
Step 5	**ip http secure-port** *port-number* **Example:** `Device(config)# ip http secure-port 443`	(Optional) Specifies the port number to be used for the HTTPS server. The default port number is 443. Valid options are 443 or any number in the range 1025 to 65535.
Step 6	**ip http secure-ciphersuite** {[**3des-ede-cbc-sha**] [**rc4-128-md5**] [**rc4-128-sha**] [**des-cbc-sha**]} **Example:** `Device(config)# ip http secure-ciphersuite rc4-128-md5`	(Optional) Specifies the CipherSuites (encryption algorithms) to be used for encryption over the HTTPS connection. If you do not have a reason to specify a particularly CipherSuite, you should allow the server and client to negotiate a CipherSuite that they both support. This is the default.
Step 7	**ip http secure-client-auth** **Example:** `Device(config)# ip http secure-client-auth`	(Optional) Configures the HTTP server to request an X.509v3 certificate from the client for authentication during the connection process. The default is for the client to request a certificate from the server, but the server does not attempt to authenticate the client.
Step 8	**ip http secure-trustpoint** *name* **Example:** `Device(config)# ip http secure-trustpoint your_trustpoint`	Specifies the CA trustpoint to use to get an X.509v3 security certificate and to authenticate the client certificate connection. **Note** Use of this command assumes you have already configured a CA trustpoint according to the previous procedure.
Step 9	**ip http path** *path-name* **Example:** `Device(config)# ip http path /your_server:80`	(Optional) Sets a base HTTP path for HTML files. The path specifies the location of the HTTP server files on the local system (usually located in system flash memory).
Step 10	**ip http access-class** { **ipv4** {*access-list-number* | *access-list-name*} | **ipv6** {*access-list-name*} }	(Optional)Specifies an access list to use to allow access to the HTTP server.

	Command or Action	Purpose
	Example: `Device(config)# ip http access-class ipv4 4`	
Step 11	**ip http max-connections** *value* **Example:** `Device(config)# ip http max-connections 4`	(Optional) Sets the maximum number of concurrent connections that are allowed to the HTTP server. We recommend that the value be at least 10 and not less. This is required for the UI to function as expected.
Step 12	**ip http timeout-policy idle** *seconds* **life** *seconds* **requests** *value* **Example:** `Device(config)# ip http timeout-policy idle 120 life 240 requests 1`	(Optional) Specifies how long a connection to the HTTP server can remain open under the defined circumstances: • **idle**: the maximum time period when no data is received or response data cannot be sent. The range is 1 to 600 seconds. The default is 180 seconds (3 minutes). • **life**: the maximum time period from the time that the connection is established. The range is 1 to 86400 seconds (24 hours). The default is 180 seconds. • **requests**: the maximum number of requests processed on a persistent connection. The maximum value is 86400. The default is 1.
Step 13	**end** **Example:** `Device(config)#end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring the Secure HTTP Client

Beginning in privileged EXEC mode, follow these steps to configure a secure HTTP client:

Before you begin

The standard HTTP client and secure HTTP client are always enabled. A certificate authority is required for secure HTTP client certification. This procedure assumes that you have previously configured a CA trustpoint on the switch. If a CA trustpoint is not configured and the remote HTTPS server requires client authentication, connections to the secure HTTP client fail.

Procedure

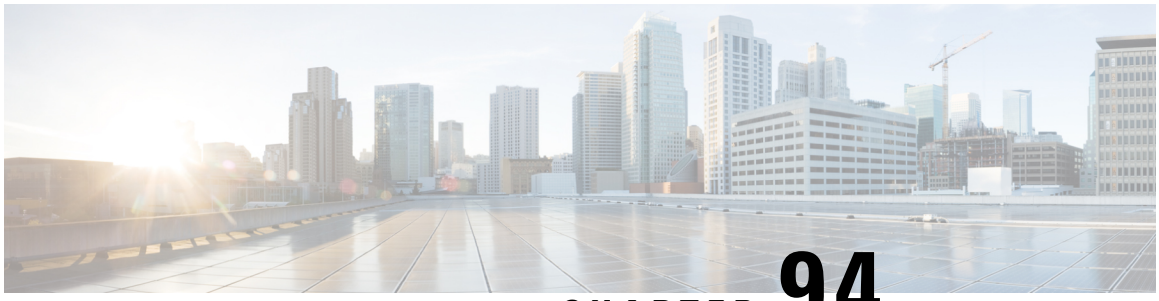
	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip http client secure-trustpoint** *name* **Example:** `Device(config)# ip http client secure-trustpoint your_trustpoint`	(Optional) Specifies the CA trustpoint to be used if the remote HTTP server requests client authentication. Using this command assumes that you have already configured a CA trustpoint by using the previous procedure. The command is optional if client authentication is not needed or if a primary trustpoint has been configured.
Step 4	**ip http client secure-ciphersuite** {[**3des-ede-cbc-sha**] [**rc4-128-md5**] [**rc4-128-sha**] [**des-cbc-sha**]} **Example:** `Device(config)# ip http client secure-ciphersuite rc4-128-md5`	(Optional) Specifies the CipherSuites (encryption algorithms) to be used for encryption over the HTTPS connection. If you do not have a reason to specify a particular CipherSuite, you should allow the server and client to negotiate a CipherSuite that they both support. This is the default.
Step 5	**end** **Example:** `Device(config)#end`	Exits global configuration mode and returns to privileged EXEC mode.

Monitoring Secure HTTP Server and Client Status

Table 102: Commands for Displaying the SSL Secure Server and Client Status

Command	Purpose
show ip http client secure status	Shows the HTTP secure client configuration.
show ip http server secure status	Shows the HTTP secure server configuration.
show running-config	Shows the generated self-signed certificate for secure HTTP connections.



CHAPTER 94

IPv4 ACLs

Restrictions for IPv4 Access Control Lists

General Network Security

The following are restrictions for configuring network security with ACLs:

- Not all commands that accept a numbered ACL accept a named ACL. ACLs for packet filters and route filters on interfaces can use a name. VLAN maps also accept a name.
- A standard ACL and an extended ACL cannot have the same name.
- Though visible in the command-line help strings, **appletalk** is not supported as a matching condition for the **deny** and **permit** MAC access-list configuration mode commands.
- ACLs cannot be configured on management ports.
- ACL wildcard is not supported in downstream client policy.
- When you apply a scale ACL to an interface that does not program TCAM for a protocol and the ACLs that have been unloaded, it can impact the existing normal movement of traffic for other protocols. The restriction is applicable to IPv6 and MAC address traffic.
- Router ACL is enforced on all types of traffic, including CPU generated traffic.
- ACL logging in the egress direction are not supported for packets that are generated from the control plane of the device.
- Time-to-live (TTL) classification is not supported on ACLs.
- If a downloadable ACL contains any type of duplicate entries, the entries are not auto merged. As a result, the 802.1X session authorization fails. Ensure that the downloadable ACL is optimized without any duplicate entries, for example port-based and name-based entries for the same port.
- Egress ACL lookup is not supported for injected traffic that is forwarded by the software.

IPv4 ACL Network Interfaces

The following restrictions apply to IPv4 ACLs to network interfaces:

- When controlling access to an interface, you can use a named or numbered ACL.
- If you apply an ACL to a Layer 2 interface that is a member of a VLAN, the Layer 2 (port) ACL takes precedence over an input Layer 3 ACL applied to the VLAN interface or a VLAN map applied to the VLAN.
- If you apply an ACL to a Layer 3 interface and routing is not enabled on the switch, the ACL only filters packets that are intended for the CPU, such as SNMP, Telnet, or web traffic.
- If the **preauth_ipv4_acl** ACL is configured to filter packets, the ACL is removed after authentication.
- You do not have to enable routing to apply ACLs to Layer 2 interfaces.

MAC ACLs on a Layer 2 Interface

After you create a MAC ACL, you can apply it to a Layer 2 interface to filter non-IP traffic coming in that interface. When you apply the MAC ACL, consider these guidelines:

- You can apply no more than one IP access list and one MAC access list to the same Layer 2 interface. The IP access list filters only IP packets, and the MAC access list filters non-IP packets.
- A Layer 2 interface can have only one MAC access list. If you apply a MAC access list to a Layer 2 interface that has a MAC ACL configured, the new ACL replaces the previously configured one.

Note The **mac access-group** interface configuration command is only valid when applied to a physical Layer 2 interface. You cannot use the command on EtherChannel port channels.

IP Access List Entry Sequence Numbering

- This feature does not support dynamic, reflexive, or firewall access lists.

Information About IPv4 Access Control Lists

ACL Overview

Packet filtering can help limit network traffic and restrict network use by certain users or devices. ACLs filter traffic as it passes through a device and permit or deny packets crossing specified interfaces. An ACL is a sequential collection of permit and deny conditions that apply to packets. When a packet is received on an interface, the switch compares the fields in the packet against any applied ACLs to verify that the packet has the required permissions to be forwarded, based on the criteria specified in the access lists. One by one, it tests packets against the conditions in an access list. The first match decides whether the switch accepts or rejects the packets. Because the switch stops testing after the first match, the order of conditions in the list is critical. If no conditions match, the switch rejects the packet. If there are no restrictions, the switch forwards the packet; otherwise, the switch drops the packet. The switch can use ACLs on all packets it forwards.

You configure access lists on a device to provide basic security for your network. If you do not configure ACLs, all packets passing through the switch could be allowed onto all parts of the network. You can use ACLs to control which hosts can access different parts of a network or to decide which types of traffic are forwarded or blocked at device interfaces. For example, you can allow e-mail traffic to be forwarded but not Telnet traffic.

Access Control Entries

An ACL contains an ordered list of access control entries (ACEs). Each ACE specifies *permit* or *deny* and a set of conditions the packet must satisfy in order to match the ACE. The meaning of *permit* or *deny* depends on the context in which the ACL is used.

ACL Supported Types

The device supports IP ACLs and Ethernet (MAC) ACLs:

- IP ACLs filter IPv4 traffic, including TCP, User Datagram Protocol (UDP), Internet Group Management Protocol (IGMP), and Internet Control Message Protocol (ICMP).
- Ethernet ACLs filter non-IP traffic.

This device also supports quality of service (QoS) classification ACLs.

Supported ACLs

The switch supports three types of ACLs to filter the traffic:

- Port ACLs access-control traffic entering a Layer 2 interface. You can apply port ACLs to a Layer 2 interface in each direction to each access list type—IPv4, IPv6, and MAC.
- Router ACLs access-control traffic routed between VLANs and are applied to Layer 3 interfaces in a specific direction (inbound or outbound).

ACL Precedence

When VLAN maps, Port ACLs, and router ACLs are configured on the same switch, the filtering precedence, from greatest to least for ingress traffic is port ACL, VLAN map, and then router ACL. For egress traffic, the filtering precedence is router ACL, VLAN map, and then port ACL.

The following examples describe simple use cases:

- When both an input port ACL and a VLAN map are applied, incoming packets received on ports with a port ACL applied are filtered by the port ACL. Other packets are filtered by the VLAN map
- When an input router ACL and input port ACL exist in a switch virtual interface (SVI), incoming packets received on ports to which a port ACL is applied are filtered by the port ACL. Incoming routed IP packets received on other ports are filtered by the router ACL. Other packets are not filtered.
- When an output router ACL and input port ACL exist in an SVI, incoming packets received on the ports to which a port ACL is applied are filtered by the port ACL. Outgoing routed IP packets are filtered by the router ACL. Other packets are not filtered.

- When a VLAN map, input router ACL, and input port ACL exist in an SVI, incoming packets received on the ports to which a port ACL is applied are only filtered by the port ACL. Incoming routed IP packets received on other ports are filtered by both the VLAN map and the router ACL. Other packets are filtered only by the VLAN map.
- When a VLAN map, output router ACL, and input port ACL exist in an SVI, incoming packets received on the ports to which a port ACL is applied are only filtered by the port ACL. Outgoing routed IP packets are filtered by both the VLAN map and the router ACL. Other packets are filtered only by the VLAN map.

Port ACLs

Port ACLs are ACLs that are applied to Layer 2 interfaces on a switch. Port ACLs are supported on physical interfaces and EtherChannel interfaces but not on EtherChannel member interfaces. Port ACLs can be applied to the interface in inbound and outbound direction. The following access lists are supported:

- Standard IP access lists using source addresses
- Extended IP access lists using source and destination addresses and optional protocol type information
- MAC extended access lists using source and destination MAC addresses and optional protocol type information

The switch examines ACLs on an interface and permits or denies packet forwarding based on how the packet matches the entries in the ACL. In this way, ACLs control access to a network or to part of a network.

Figure 91: Using ACLs to Control Traffic in a Network

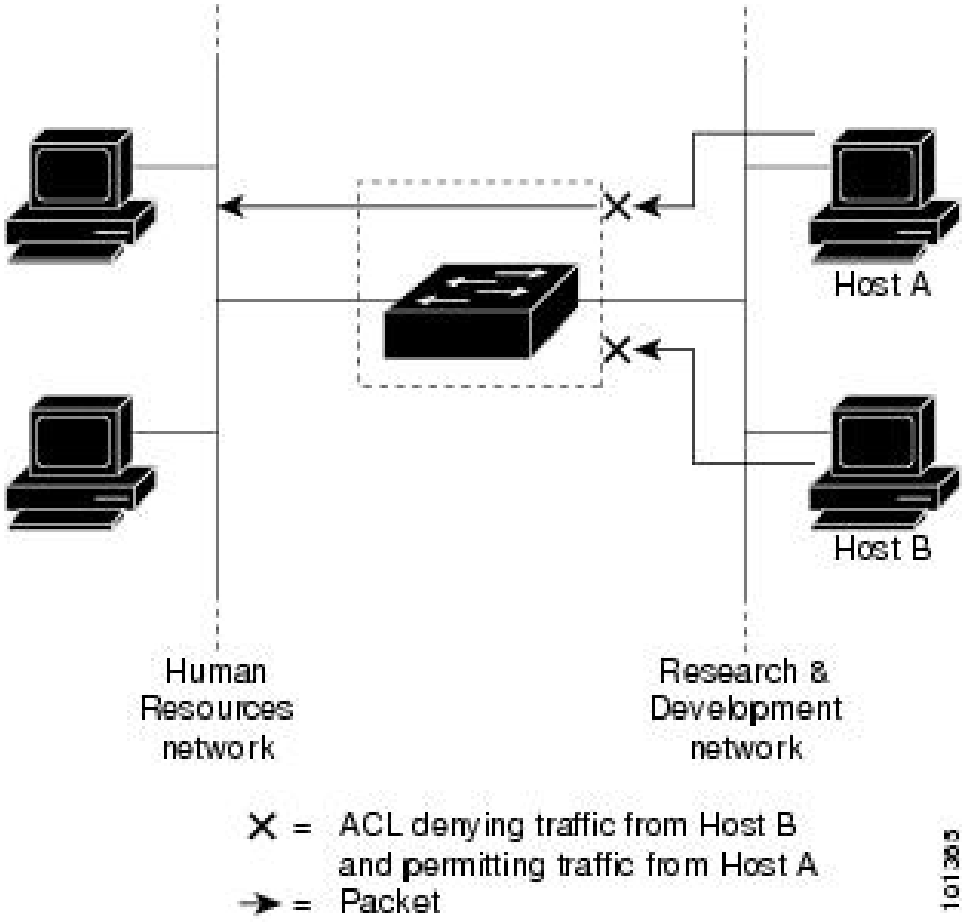


Figure 92: Using ACLs to Control Traffic in a Network

This is an example of using port ACLs to control access to a network when all workstations are in the same VLAN. ACLs applied at the Layer 2 input would allow Host A to access the Human Resources network, but prevent Host B from accessing the same network. Port ACLs can only be applied to Layer 2 interfaces in the inbound direction.

When you apply a port ACL to a trunk port, the ACL filters traffic on all VLANs present on the trunk port. When you apply a port ACL to a port with voice VLAN, the ACL filters traffic on both data and voice VLANs.

With port ACLs, you can filter IP traffic by using IP access lists and non-IP traffic by using MAC addresses. You can filter both IP and non-IP traffic on the same Layer 2 interface by applying both an IP access list and a MAC access list to the interface.

Note You can't apply more than one IP access list and one MAC access list to a Layer 2 interface. If an IP access list or MAC access list is already configured on a Layer 2 interface and you apply a new IP access list or MAC access list to the interface, the new ACL replaces the previously configured one.

Router ACLs

You can apply router ACLs on switch virtual interfaces (SVIs), which are Layer 3 interfaces to VLANs; on physical Layer 3 interfaces; and on Layer 3 EtherChannel interfaces. You apply router ACLs on interfaces for specific directions (inbound or outbound). You can apply one router ACL in each direction on an interface.

The switch supports these access lists for IPv4 traffic:

- Standard IP access lists use source addresses for matching operations.
- Extended IP access lists use source and destination addresses and optional protocol type information for matching operations.

As with port ACLs, the switch examines ACLs associated with features configured on a given interface. As packets enter the switch on an interface, ACLs associated with all inbound features configured on that interface are examined. After packets are routed and before they are forwarded to the next hop, all ACLs associated with outbound features configured on the egress interface are examined.

ACLs permit or deny packet forwarding based on how the packet matches the entries in the ACL, and can be used to control access to a network or to part of a network.

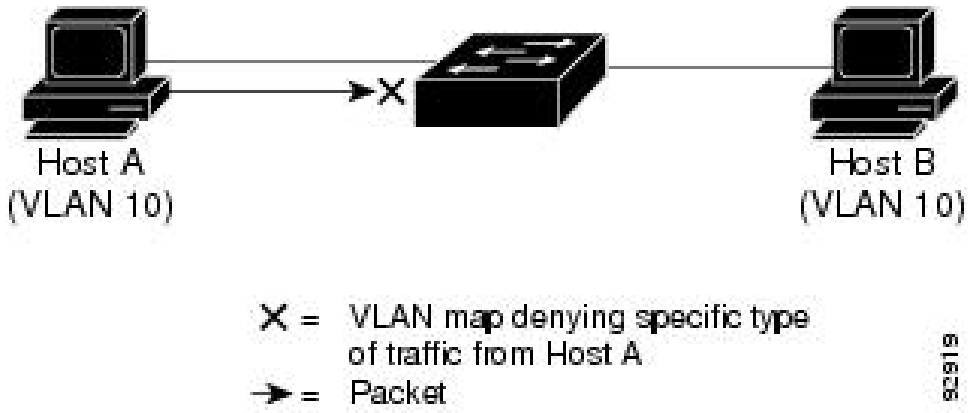
VLAN Maps

VLAN ACLs or VLAN maps are used to control the network traffic within a VLAN. You can apply VLAN maps to all packets that are bridged within a VLAN in the switch. VACLs are strictly for the security packet filtering and for redirecting traffic to specific physical interfaces. VACLs are not defined by direction (ingress or egress).

All non-IP protocols are access-controlled through MAC addresses and Ethertype using MAC VLAN maps. (IP traffic is not access-controlled by MAC VLAN maps.) You can enforce VLAN maps only on packets going through the switch; you cannot enforce VLAN maps on traffic between hosts on a hub or on another switch that is connected to this switch.

With VLAN maps, forwarding of packets is permitted or denied, based on the action specified in the map.

Figure 93: Using VLAN Maps to Control Traffic



ACEs and Fragmented and Unfragmented Traffic

IP packets can be fragmented as they cross the network. When this happens, only the fragment containing the beginning of the packet contains the Layer 4 information, such as TCP or UDP port numbers, ICMP type and code, and so on. All other fragments are missing this information.

Some access control entries (ACEs) do not check Layer 4 information and therefore can be applied to all packet fragments. ACEs that do test Layer 4 information cannot be applied in the standard manner to most of the fragments in a fragmented IP packet. When the fragment contains no Layer 4 information and the ACE tests some Layer 4 information, the matching rules are modified:

- Permit ACEs that check the Layer 3 information in the fragment (including protocol type, such as TCP, UDP, and so on) are considered to match the fragment regardless of what the missing Layer 4 information might have been.

Note For TCP ACEs with L4 Ops, the fragmented packets will be dropped per RFC 1858.

- Deny ACEs that check Layer 4 information never match a fragment unless the fragment contains Layer 4 information.

Standard and Extended IPv4 ACLs

An ACL is a sequential collection of permit and deny conditions. One by one, the device tests packets against the conditions in an access list. The first match determines whether the device accepts or rejects the packet. Because the device stops testing after the first match, the order of the conditions is critical. If no conditions match, the device denies the packet.

The software supports these types of ACLs or access lists for IPv4:

- Standard IP access lists use source addresses for matching operations.
- Extended IP access lists use source and destination addresses for matching operations and optional protocol-type information for finer granularity of control.

IPv4 ACL Switch Unsupported Features

The following ACL-related features are not supported:

- Non-IP protocol ACLs
- IP accounting
- Reflexive ACLs and dynamic ACLs are not supported.

Numbered Standard IPv4 ACLs

When creating an ACL, remember that, by default, the end of the ACL contains an implicit deny statement for all packets that it did not find a match for before reaching the end. With standard access lists, if you omit the mask from an associated IP host address ACL specification, 0.0.0.0 is assumed to be the mask.

When using the **show ip access-list** *acl_name* or the **show run section** *acl_name* command, the ACEs are displayed in ascending order according to their sequence numbers.

After creating a numbered standard IPv4 ACL, you can apply it to VLANs, to terminal lines, or to interfaces.

Numbered Extended IPv4 ACLs

Although standard ACLs use only source addresses for matching, you can use extended ACL source and destination addresses for matching operations and optional protocol type information for finer granularity of control. When you are creating ACEs in numbered extended access lists, remember that after you create the ACL, any additions are placed at the end of the list. You cannot reorder the list or selectively add or remove ACEs from a numbered list.

The device does not support dynamic or reflexive access lists. It also does not support filtering based on the type of service (ToS) minimize-monetary-cost bit.

Some protocols also have specific parameters and keywords that apply to that protocol.

You can define an extended TCP, UDP, ICMP, IGMP, or other IP ACL. The device also supports these IP protocols:

Note ICMP echo-reply cannot be filtered. All other ICMP codes or types can be filtered.

These IP protocols are supported:

- Authentication Header Protocol (**ahp**)
- Encapsulation Security Payload (**esp**)
- Enhanced Interior Gateway Routing Protocol (**eigrp**)
- Generic routing encapsulation (**gre**)
- Internet Control Message Protocol (**icmp**)
- Internet Group Management Protocol (**igmp**)
- Any Interior Protocol (**ip**)
- IP in IP tunneling (**ipinip**)
- KA9Q NOS-compatible IP over IP tunneling (**nos**)
- Open Shortest Path First routing (**ospf**)
- Payload Compression Protocol (**pcp**)
- Protocol-Independent Multicast (**pim**)
- Transmission Control Protocol (**tcp**)
- User Datagram Protocol (**udp**)

Named IPv4 ACLs

You can identify IPv4 ACLs with an alphanumeric string (a name) rather than a number. You can use named ACLs to configure more IPv4 access lists in a device than if you were to use numbered access lists. If you

identify your access list with a name rather than a number, the mode and command syntax are slightly different. However, not all commands that use IP access lists accept a named access list.

Note The name you give to a standard or extended ACL can also be a number in the supported range of access list numbers. That is, the name of a standard IP ACL can be 1 to 99. The advantage of using named ACLs instead of numbered lists is that you can delete individual entries from a named list.

Consider these guidelines before configuring named ACLs:

- Numbered ACLs are also available.
- A standard ACL and an extended ACL cannot have the same name.
- You can use standard or extended ACLs (named or numbered) in VLAN maps.

ACL Logging

The device software can provide logging messages about packets permitted or denied by a standard IP access list. That is, any packet that matches the ACL causes an informational logging message about the packet to be sent to the console. The level of messages logged to the console is controlled by the **logging console** commands controlling the syslog messages.

Note ACL logging is not supported for ACLs used with Unicast Reverse Path Forwarding (uRPF). It is only supported for router ACL.

Note Because routing is done in hardware and logging is done in software, if a large number of packets match a *permit* or *deny* ACE containing a **log** keyword, the software might not be able to match the hardware processing rate, and not all packets will be logged.

The first packet that triggers the ACL causes a logging message right away, and subsequent packets are collected over 5-minute intervals before they appear or logged. The logging message includes the access list number, whether the packet was permitted or denied, the source IP address of the packet, and the number of packets from that source permitted or denied in the prior 5-minute interval.

Note The logging facility might drop some logging message packets if there are too many to be handled or if there is more than one logging message to be handled in 1 second. This behavior prevents the device from crashing due to too many logging packets. Therefore, the logging facility should not be used as a billing tool or an accurate source of the number of matches to an access list.

Hardware and Software Treatment of IP ACLs

ACL processing is performed in hardware. If the hardware reaches its capacity to store ACL configurations, all packets on that interface are dropped.

Note If an ACL configuration cannot be implemented in hardware due to an out-of-resource condition on a device, then only the traffic in that VLAN arriving on that device is affected.

For router ACLs, other factors can cause packets to be sent to the CPU:

- Using the **log** keyword
- Generating ICMP unreachable messages

When you enter the **show ip access-lists** privileged EXEC command, the match count displayed does not account for packets that are access controlled in hardware. Use the **show platform software fed switch** { *switch_num* | **active** | **standby** } **acl counters hardware** privileged EXEC command to obtain some basic hardware ACL statistics for switched and routed packets.

Router ACLs function as follows:

- The hardware controls permit and deny actions of standard and extended ACLs (input and output) for security access control.
- If **log** has not been specified, the flows that match a *deny* statement in a security ACL are dropped by the hardware if *ip unreachables* is disabled. The flows matching a *permit* statement are switched in hardware.
- Adding the **log** keyword to an ACE in a router ACL causes a copy of the packet to be sent to the CPU for logging only. If the ACE is a *permit* statement, the packet is still switched and routed in hardware.

VLAN Map Configuration Guidelines

VLAN maps are the only way to control filtering within a VLAN. VLAN maps have no direction. To filter traffic in a specific direction by using a VLAN map, you need to include an ACL with specific source or destination addresses. If there is a match clause for that type of packet (IP or MAC) in the VLAN map, the default action is to drop the packet if the packet does not match any of the entries within the map. If there is no match clause for that type of packet, the default is to forward the packet.

The following are the VLAN map configuration guidelines:

- If there is no ACL configured to deny traffic on an interface and no VLAN map is configured, all traffic is permitted.
- Each VLAN map consists of a series of entries. The order of entries in an VLAN map is important. A packet that comes into the device is tested against the first entry in the VLAN map. If it matches, the action specified for that part of the VLAN map is taken. If there is no match, the packet is tested against the next entry in the map.
- If the VLAN map has at least one match clause for the type of packet (IP or MAC) and the packet does not match any of these match clauses, the default is to drop the packet. If there is no match clause for that type of packet in the VLAN map, the default is to forward the packet.
- Logging is not supported for VLAN maps.
- When a device has an IP access list or MAC access list applied to a Layer 2 interface, and you apply a VLAN map to a VLAN that the port belongs to, the port ACL takes precedence over the VLAN map.

- If a VLAN map configuration cannot be applied in hardware, all packets in that VLAN are dropped.

VLAN Maps with Router ACLs

To access control both bridged and routed traffic, you can use VLAN maps only or a combination of router ACLs and VLAN maps. You can define router ACLs on both input and output routed VLAN interfaces, and you can define a VLAN map to access control the bridged traffic.

If a packet flow matches a VLAN-map deny clause in the ACL, regardless of the router ACL configuration, the packet flow is denied.

Note When you use router ACLs with VLAN maps, packets that require logging on the router ACLs are not logged if they are denied by a VLAN map.

If the VLAN map has a match clause for the type of packet (IP or MAC) and the packet does not match the type, the default is to drop the packet. If there is no match clause in the VLAN map, and no action specified, the packet is forwarded if it does not match any VLAN map entry.

VLAN Maps and Router ACL Configuration Guidelines

These guidelines are for configurations where you need to have an router ACL and a VLAN map on the same VLAN. These guidelines do not apply to configurations where you are mapping router ACLs and VLAN maps on different VLANs.

If you must configure a router ACL and a VLAN map on the same VLAN, use these guidelines for both router ACL and VLAN map configuration:

- You can configure only one VLAN map and one router ACL in each direction (input/output) on a VLAN interface.
- Whenever possible, try to write the ACL with all entries having a single action except for the final, default action of the other type. That is, write the ACL using one of these two forms:
  - permit...

    permit...

    permit...

    deny ip any any

  or

  - deny...

    deny...

    deny...

    permit ip any any

- To define multiple actions in an ACL (permit, deny), group each action type together to reduce the number of entries.

- Avoid including Layer 4 information in an ACL; adding this information complicates the merging process. The best merge results are obtained if the ACLs are filtered based on IP addresses (source and destination) and not on the full flow (source IP address, destination IP address, protocol, and protocol ports). It is also helpful to use *don't care* bits in the IP address, whenever possible.

  If you need to specify the full-flow mode and the ACL contains both IP ACEs and TCP/UDP/ICMP ACEs with Layer 4 information, put the Layer 4 ACEs at the end of the list. This gives priority to the filtering of traffic based on IP addresses.

Time Ranges for ACLs

You can selectively apply extended ACLs based on the time of day and the week by using the **time-range** global configuration command. First, define a time-range name and set the times and the dates or the days of the week in the time range. Then enter the time-range name when applying an ACL to set restrictions to the access list. You can use the time range to define when the permit or deny statements in the ACL are in effect, for example, during a specified time period or on specified days of the week. The **time-range** keyword and argument are referenced in the named and numbered extended ACL task tables.

These are some benefits of using time ranges:

- You have more control over permitting or denying a user access to resources, such as an application (identified by an IP address/mask pair and a port number).
- You can control logging messages. ACL entries can be set to log traffic only at certain times of the day. Therefore, you can simply deny access without needing to analyze many logs generated during peak hours.

Time-based access lists trigger CPU activity because the new configuration of the access list must be merged with other features and the combined configuration loaded into the hardware memory. For this reason, you should be careful not to have several access lists configured to take affect in close succession (within a small number of minutes of each other.)

Note The time range relies on the device system clock; therefore, you need a reliable clock source. We recommend that you use Network Time Protocol (NTP) to synchronize the device clock.

IPv4 ACL Interface Considerations

For inbound ACLs, after receiving a packet, the device checks the packet against the ACL. If the ACL permits the packet, the device continues to process the packet. If the ACL rejects the packet, the device discards the packet.

For outbound ACLs, after receiving and routing a packet to a controlled interface, the device checks the packet against the ACL. If the ACL permits the packet, the device sends the packet. If the ACL rejects the packet, the device discards the packet.

By default, the input interface sends ICMP Unreachable messages whenever a packet is discarded, regardless of whether the packet was discarded because of an ACL on the input interface or because of an ACL on the output interface. ICMP Unreachables are normally limited to no more than one every one-half second per input interface, but this can be changed by using the **ip icmp rate-limit unreachable** global configuration command.

When you apply an undefined ACL to an interface, the device acts as if the ACL has not been applied to the interface and permits all packets. Remember this behavior if you use undefined ACLs for network security.

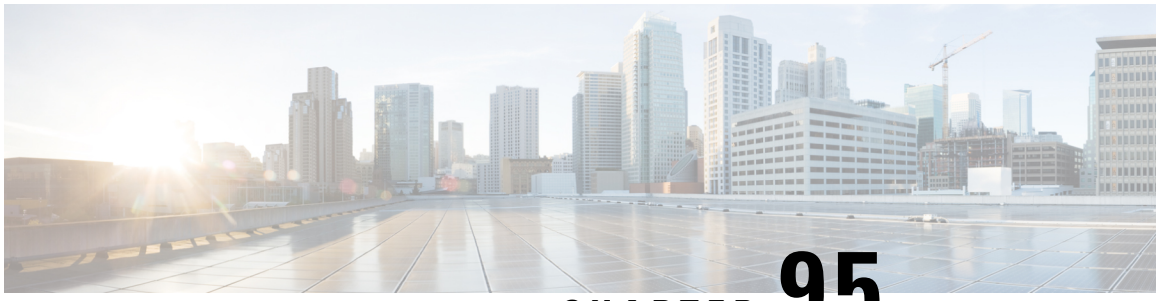
Monitoring IPv4 ACLs

You can monitor IPv4 ACLs by displaying the ACLs that are configured on the device, and displaying the ACLs that have been applied to interfaces and VLANs.

When you use the **ip access-group** interface configuration command to apply ACLs to a Layer 2 or 3 interface, you can display the access groups on the interface. You can also display the MAC ACLs applied to a Layer 2 interface. You can use the privileged EXEC commands as described in this table to display this information.

Table 103: Commands for Displaying Access Lists and Access Groups

Command	Purpose
show access-lists [*number* | *name*]	Displays the contents of one or all current IP and MAC address access lists or a specific access list (numbered or named).
show ip access-lists [*number* | *name*]	Displays the contents of all current IP access lists or a specific IP access list (numbered or named).
show ip interface *interface-id*	Displays detailed configuration and status of an interface. If IP is enabled on the interface and ACLs have been applied by using the **ip access-group** interface configuration command, the access groups are included in the display.
show running-config [**interface** *interface-id*]	Displays the contents of the configuration file for the device or the specified interface, including all configured MAC and IP access lists and which access groups are applied to an interface.
show mac access-group [**interface** *interface-id*]	Displays MAC access lists applied to all Layer 2 interfaces or the specified Layer 2 interface.



CHAPTER 95

IPv6 ACLs

Restrictions for IPv6 ACLs

IPv6 supports only named ACLs. With IPv4 ACLs, you can configure standard and extended numbered IP ACLs, named IP ACLs, and MAC ACLs.

The switch supports most Cisco IOS-supported IPv6 ACLs with some exceptions:

- The switch does not support matching on these keywords: **flowlabel**, **routing header**, and **undetermined-transport**.
- The switch does not support reflexive ACLs (the **reflect** keyword).
- The **vrf-also** keyword is mutually exclusive of IPv6 access-class line command.
- The switch does not apply MAC-based ACLs on IPv6 frames.
- When configuring an ACL, there is no restriction on keywords that are entered in the ACL, regardless of whether they are supported or not on the platform. When you apply the ACL to an interface that requires hardware forwarding (physical ports or SVIs), the switch checks to determine whether ACL can be supported on the interface or not. If the ACL is not supported on the interface, the ACL is rejected.
- If an ACL is applied to an interface and you attempt to add an access control entry (ACE) with an unsupported keyword, the switch does not allow the ACE to be added to the ACL that is currently attached to the interface.
- When you apply a scale ACL to an interface that does not program TCAM for a protocol and the ACLs that have been unloaded, it can impact the existing normal movement of traffic for other protocols. The restriction is applicable to IPv6 and MAC address traffic.
- Time-to-live (TTL) classification is not supported on ACLs.
- If a downloadable ACL contains any type of duplicate entries, the entries are not auto merged. As a result, the 802.1X session authorization fails. Ensure that the downloadable ACL is optimized without any duplicate entries, for example port-based and name-based entries for the same port.

- Egress ACL lookup is not supported for injected traffic that is forwarded by the software.
- ACLs support only Layer 3 interfaces (such as routed interfaces and VLAN interfaces).

Information About IPv6 ACLs

The following sections provide information about IPv6 ACLs.

IPv6 ACL Overview

This topic provides an overview of IPv6 ACL.

An access control list (ACL) is a set of rules that are used to limit access to a particular interface. ACLs are configured on the device and applied to the management interface and to any of the dynamic interfaces.

You can also create a preauthentication ACL for web authentication. Such an ACL is used to allow certain types of traffic before authentication is complete.

IPv6 ACLs support the same options as IPv4 ACLs including source, destination, source, and destination ports.

Supported ACLs

The switch supports three types of ACLs to filter the traffic:

- Port ACLs access-control traffic entering a Layer 2 interface. You can apply port ACLs to a Layer 2 interface in each direction to each access list type—IPv4, IPv6, and MAC.
- Router ACLs access-control traffic routed between VLANs and are applied to Layer 3 interfaces in a specific direction (inbound or outbound).

Types of ACL

The following sections provide information on the types of ACL:

Per-User IPv6 ACL

For the per-user ACL, the full access control entries (ACE) as the text strings are configured on the Cisco Secure Access Control Server (Cisco Secure ACS).

Filter ID IPv6 ACL

For the filter-Id ACL, the full ACEs and the `acl name(filter-id)` is configured on the device and only the `filter-id` is configured on the Cisco Secure ACS.

Downloadable IPv6 ACL

For the downloadable ACL (dACL), all the full ACEs and the dacl name are configured only on the Cisco Secure ACS.

The Cisco Secure ACS sends the dacl name to the device in its `ACCESS-Accept` attribute, which takes the `dacl` name and sends the `dACL` name back to the Cisco Secure ACS for the ACEs, using the `ACCESS-request` attribute.

ACL Precedence

When Port ACLs, and router ACLs are configured on the same switch, the filtering precedence, from greatest to least for ingress traffic is port ACL, and then router ACL. For egress traffic, the filtering precedence is router ACL, and then port ACL.

The following examples describe simple use cases:

- When an input router ACL and input port ACL exist in a switch virtual interface (SVI), incoming packets that are received on ports to which a port ACL is applied are filtered by the port ACL. Incoming routed IP packets received on other ports are filtered by the router ACL. Other packets are not filtered.
- When an output router ACL and input port ACL exist in an SVI, incoming packets that are received on the ports to which a port ACL is applied are filtered by the port ACL. Outgoing routed IP packets are filtered by the router ACL. Other packets are not filtered.

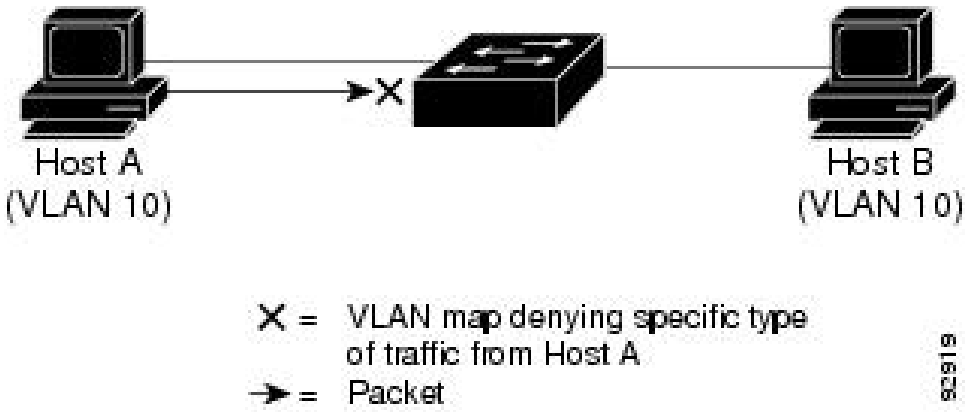
VLAN Maps

VLAN ACLs or VLAN maps are used to control the network traffic within a VLAN. You can apply VLAN maps to all packets that are bridged within a VLAN in the switch. VACLs are strictly for the security packet filtering and for redirecting traffic to specific physical interfaces. VACLs are not defined by direction (ingress or egress).

All non-IP protocols are access-controlled through MAC addresses and Ethertype using MAC VLAN maps. (IP traffic is not access-controlled by MAC VLAN maps.) You can enforce VLAN maps only on packets going through the switch; you cannot enforce VLAN maps on traffic between hosts on a hub or on another switch that is connected to this switch.

With VLAN maps, forwarding of packets is permitted or denied, based on the action specified in the map.

Figure 94: Using VLAN Maps to Control Traffic



Interactions with Other Features and Switches

- If an IPv6 router ACL is configured to deny a packet, the packet is not routed. A copy of the packet is sent to the Internet Control Message Protocol (ICMP) queue to generate an ICMP unreachable message for the frame.
- If a bridged frame is to be dropped due to a port ACL, the frame is not bridged.
- You can create both IPv4 and IPv6 ACLs on a switch, and you can apply both IPv4 and IPv6 ACLs to the same interface. Each ACL must have a unique name; an error message appears if you try to use a name that is already configured.

  You use different commands to create IPv4 and IPv6 ACLs and to attach IPv4 or IPv6 ACLs to the same Layer 2 or Layer 3 interface. If you use the wrong command to attach an ACL (for example, an IPv4 command to attach an IPv6 ACL), you receive an error message.
- You cannot use MAC ACLs to filter IPv6 frames. MAC ACLs can only filter non-IP frames.
- If the hardware memory is full, packets are dropped on the interface and an unload error message is logged.

  If the hardware memory is full, for any additional configured ACLs, packets are dropped to the CPU, and the ACLs are applied in software. When the hardware is full a message is printed to the console indicating the ACL has been unloaded and the packets will be dropped on the interface.

How to Configure an IPv6 ACL

The following sections display information on how to configure an IPv6 ACL.

Default Configuration for IPv6 ACLs

The default IPv6 ACL configuration is as follows:

```
Device# show access-lists preauth_ipv6_acl

IPv6 access list preauth_ipv6_acl (per-user)
permit udp any any eq domain sequence 10
permit tcp any any eq domain sequence 20
permit icmp any any nd-ns sequence 30
permit icmp any any nd-na sequence 40
permit icmp any any router-solicitation sequence 50
permit icmp any any router-advertisement sequence 60
permit icmp any any redirect sequence 70
permit udp any eq 547 any eq 546 sequence 80
permit udp any eq 546 any eq 547 sequence 90
deny ipv6 any any sequence 100
```

Configuring IPv6 ACLs

To filter IPv6 traffic, perform this procedure.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** {*list-name* | **log-update threshold** | **role-based** *list-name*} **Example:** `Device(config)# ipv6 access-list example_acl_list`	Defines an IPv6 ACL name, and enters IPv6 access list configuration mode.
Step 4	{**deny** | **permit**} protocol {*source-ipv6-prefix/ |prefix-length* |**any threshold**| **host** *source-ipv6-address*} [operator [*port-number*]] { *destination-ipv6-prefix/ prefix-length* | **any** | **host** *destination-ipv6-address*} [operator [*port-number*]][**dscp** *value*] [**fragments**] [**log**] [**log-input**][**sequence** *value*] [**time-range** *name*] **Example:** `Device(config-ipv6-acl)# permit tcp 2001:DB8:0300:0201::/32 eq telnet any`	Specifies permit or deny conditions for an IPv6 ACL. • For protocol, enter the name or number of an IP: **ahp**, **esp**, **icmp**, **ipv6**, **pcp**, **stcp**, **tcp**, or **udp**, or an integer in the range 0 to 255 representing an IPv6 protocol number. • The *source-ipv6-prefix/prefix-length* or *destination-ipv6-prefix/ prefix-length* is the source or destination IPv6 network or class of networks for which to set deny or permit conditions, specified in hexadecimal and using 16-bit values between colons (see RFC 2373). • Enter **any** as an abbreviation for the IPv6 prefix ::/0. • For **host** *source-ipv6-address* or *destination-ipv6-address*, enter the source or destination IPv6 host address for which to set deny or permit conditions, specified in hexadecimal using 16-bit values between colons. • (Optional) For operator, specify an operand that compares the source or destination ports of the specified protocol. Operands are **lt** (less than), **gt** (greater than), **eq** (equal), **neq** (not equal), and **range**.

	Command or Action	Purpose
		If the operator follows the *source-ipv6-prefix/prefix-length* argument, it must match the source port. If the operator follows the *destination-ipv6-prefix/prefix-length* argument, it must match the destination port. • (Optional) The **port-number** is a decimal number from 0 to 65535 or the name of a TCP or UDP port. You can use TCP port names only when filtering TCP. You can use UDP port names only when filtering UDP. • (Optional) Enter **dscp** value to match a differentiated services code point value against the traffic class value in the Traffic Class field of each IPv6 packet header. The acceptable range is from 0 to 63. • (Optional) Enter **fragments** to check noninitial fragments. This keyword is visible only if the protocol is ipv6. • (Optional) Enter **log** to cause an logging message to be sent to the console about the packet that matches the entry. Enter **log-input** to include the input interface in the log entry. Logging is supported only for router ACLs. • (Optional) Enter **sequence** *value* to specify the sequence number for the access list statement. The acceptable range is from 1 to 4,294,967,295. • (Optional) Enter **time-range** name to specify the time range that applies to the deny or permit statement.
Step 5	{**deny** | **permit**} **tcp** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [**operator** [**port-number**]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*} [operator [*port-number*]] [**ack**] [**dscp** *value*] [**established**] [**fin**] [**log**] [**log-input**] [**neq** {**port** | protocol}] [**psh**] [**range** {**port** | **protocol**}] [**rst**] [**sequence** *value*] [**syn**] [**time-range** *name*] [**urg**] **Example:**	Specifies permit or deny conditions for an IPv6 ACL. Enter **tcp** for Transmission Control Protocol. The parameters are the same as those described in Step 3a, with these additional optional parameters: • **ack**: Acknowledgment bit set. • **established**: An established connection. A match occurs if the TCP datagram has the ACK or RST bits set.

	Command or Action	Purpose
	`Device(config-ipv6-acl)# deny tcp host 2001:DB8:1::1 any log-input`	• **fin**: Finished bit set; no more data from sender. • **neq** { *port* |**protocol**}: Matches only packets that are not on a given port number. • **psh**: Push function bit set. • **range** { *port* |**protocol**}: Matches only packets in the port number range. • **rst**: Reset bit set. • **syn**: Synchronize bit set. • **urg**: Urgent pointer bit set.
Step 6	**end** **Example:** `Device(config-ipv6-acl)# end`	Exits IPv6 access list configuration mode and returns to privileged EXEC mode.
Step 7	**show ipv6 access-list** **Example:** `Device# show ipv6 access-list`	Verifies that IPv6 ACLs are configured correctly.

Attaching an IPv6 ACL to an Interface

You can apply an ACL to outbound or inbound traffic on Layer 3 interfaces, or to inbound traffic on Layer 2 interfaces. You can also apply ACLs only to inbound management traffic on Layer 3 interfaces.

Follow these steps to control access to an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:**	Identifies a Layer 2 interface (for port ACLs) or Layer 3 interface (for router ACLs) on which to apply an access list, and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet 1/1`	
Step 4	**no switchport** **Example:** `Device(config-if)# no switchport`	Returns the interface to the routed-interface status and erases all further Layer 2 configuration.
Step 5	**ipv6 address** *ipv6-address* **Example:** `Device(config-if)# ipv6 address 2001:DB8::1`	Configures an IPv6 address on a Layer 3 interface (for router ACLs).
Step 6	**ipv6 traffic-filter** *access-list-name* {**in** | **out**} **Example:** `Device(config-if)# ipv6 traffic-filter acl1 in`	Applies the access list to incoming or outgoing traffic on the interface.
Step 7	**end** **Example:** `Device(config-ipv6-acl)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring an IPv6 ACL in Template Mode

Note You can configure **ipv6 traffic-filter** command in the template configuration mode. You can configure the **source template** command only once to an interface.

Beginning in privileged EXEC mode, follow these steps to configure ACL in a template:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** {*list-name* | **log-update threshold** | **role-based** *list-name*} **Example:**	Defines an IPv6 ACL name, and enters IPv6 access list configuration mode.

	Command or Action	Purpose
	`Device(config)# ipv6 access-list v6acl10`	
Step 4	**ipv6 access-list** {*list-name* | **log-update threshold** | **role-based** *list-name*} **Example:** `Device(config-ipv6-acl)#ipv6 access-list v6acl11`	Defines an IPv6 ACL name, and enters IPv6 access list configuration mode.
Step 5	**exit** **Example:** `Device(config-ipv6-acl)#exit`	Exits access-list configuration mode.
Step 6	**template** **Example:** `Device(config)# template test`	Creates a user template and enters template configuration mode.
Step 7	**ipv6 traffic-filter** {*access-list-number* | *name*} {**in** | **out**} **Example:** `Device(config-template)# ipv6 traffic-filter v6acl10 in`	Controls access to the specified interface. Enter *access-list-number* to define the access list. The access list can be a number. Enter *name* to define the access list. The access list can be a name. Enter **in** to direct the access list in the incoming direction of the interface. Enter **out** to direct the access list in the outgoing direction of the interface.
Step 8	**exit** **Example:** `Device(config-template)# exit`	Exits template configuration mode and returns to privileged EXEC mode.
Step 9	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Identifies a specific interface for configuration, and enters interface configuration mode. The interface can be a Layer 2 interface (port ACL), or a Layer 3 interface (router ACL).
Step 10	**ipv6 traffic-filter** {*access-list-number* | *name*} {**in** | **out**} **Example:** `Device(config-if)# ipv6 traffic-filter v6acl11 out`	Controls access to the specified interface. Enter *access-list-number* to define the access list. The access list can be a number. Enter *name* to define the access list. The access list can be a name. Enter **in** to direct the access list in the incoming direction of the interface. Enter **out** to direct the access list in the outgoing direction of the interface.

	Command or Action	Purpose
Step 11	**source template** *name* **Example:** `Device(config)# source template test`	Applies an interface template to a target. The accesss list *v6acl10* is the incoming access list that is configured.
Step 12	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring a VLAN Map

To create a VLAN map and apply it to one or more VLANs, perform these steps:

Before you begin

Create the IPv6 ACL that you want to apply to the VLAN.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan access-map** *name* [**number**] **Example:** `Device(config)# vlan access-map map_1 20`	Creates a VLAN map, and enters VLAN access-map command mode VLAN map can have a name or (optionally) a number. The number is the sequence number of the entry within the map. When you create VLAN maps with the same name, numbers are assigned sequentially in increments of 10. When modifying or deleting maps, you can enter the number of the map entry that you want to modify or delete. VLAN maps do not use the specific permit or deny keywords. To deny a packet by using VLAN maps, create an ACL that would match the packet, and set the action to drop. A permit in the ACL counts as a match. A deny in the ACL means no match.

	Command or Action	Purpose
Step 4	**match** {**ip** | **ipv6** | **mac**} **address** {*name* | *number*} [*name* | *number*] **Example:** `Device(config-access-map)# match ipv6 address ip_net`	Matches the packet against one or more access lists. Note that packets are only matched against access lists of the correct protocol type. IP packets are matched against IP access lists. Non-IP packets are only matched against named MAC access lists. **Note** If the VLAN map is configured with a match clause for a type of packet (IP or MAC) and the map action is drop, all packets that match the type are dropped. If the VLAN map has no match clause, and the configured action is drop, all IP and Layer 2 packets are dropped.
Step 5	Enter one of the following commands to specify an IP packet or a non-IP packet (with only a known MAC address) and to match the packet against one or more ACLs: • **action** { **forward**} `Device(config-access-map)# action forward` • **action** { **drop**} `Device(config-access-map)# action drop`	Sets the action for the map entry.
Step 6	**vlan filter** *mapname* **vlan-list** *list* **Example:** `Device(config)# vlan filter map 1 vlan-list 20-22`	Applies the VLAN map to one or more VLAN IDs. The list can be a single VLAN ID (22), a consecutive list (10-22), or a string of VLAN IDs (12, 22, 30). Spaces around the comma and hyphen are optional.
Step 7	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Applying a VLAN Map to a VLAN

To apply a VLAN map to one or more VLANs, perform these steps.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan filter** *mapname* **vlan-list** *list* **Example:** `Device(config)# vlan filter map 1 vlan-list 20-22`	Applies the VLAN map to one or more VLAN IDs. The list can be a single VLAN ID (22), a consecutive list (10-22), or a string of VLAN IDs (12, 22, 30). Spaces around the comma and hyphen are optional.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Monitoring IPv6 ACLs

You can display information about all configured access lists, all IPv6 access lists, or a specific access list by using one or more of the privileged EXEC commands shown in the table below:

Table 104: show ACL commands

Command	Purpose
show access-lists	Displays all access lists configured on the switch.
show ipv6 access-list [*access-list-name*]	Displays all configured IPv6 access lists or the access list specified by name.
show vlan access-map [*map-name*]	Displays VLAN access map configuration.

Configuration Examples for IPv6 ACL

This following sections display configuration examples for IPv6 ACL.

Example: Creating an IPv6 ACL

This example configures the IPv6 access list named IPv6-ACL. The first deny entry in the list denies all packets that have a destination TCP port number greater than 5000. The second deny entry denies packets that have a source UDP port number less than 5000. The second deny also logs all matches to the console. The first permit entry in the list permits all ICMP packets. The second permit entry in the list permits all other

traffic. The second permit entry is necessary because an implicit deny -all condition is at the end of each IPv6 access list.

Note Logging is supported only on Layer 3 interfaces.

```
Device> enable
Device(config)# ipv6 access-list IPv6_ACL
Device(config-ipv6-acl)# deny tcp any any gt 5000
Device (config-ipv6-acl)# deny ::/0 lt 5000 ::/0 log
Device(config-ipv6-acl)# permit icmp any any
Device(config-ipv6-acl)# permit any any
Device(config-ipv6-acl)# end
```

Example: Displaying IPv6 ACLs

The following is a sample output from the **show access-lists** command. The output shows all access lists that are configured on the device.

```
Device# show access-lists

Extended IP access list hello
10 permit ip any any
IPv6 access list ipv6
permit ipv6 any any sequence 10
```

The following is a sample output from the **show ipv6 access-lists** command. The output shows only IPv6 access lists configured on the switch.

```
Device# show ipv6 access-list

IPv6 access list inbound
permit tcp any any eq bgp (8 matches) sequence 10
permit tcp any any eq telnet (15 matches) sequence 20
permit udp any any sequence 30
IPv6 access list outbound
deny udp any any sequence 10
deny tcp any any eq telnet sequence 20
```

Example: Displaying VLAN Access Map Configuration

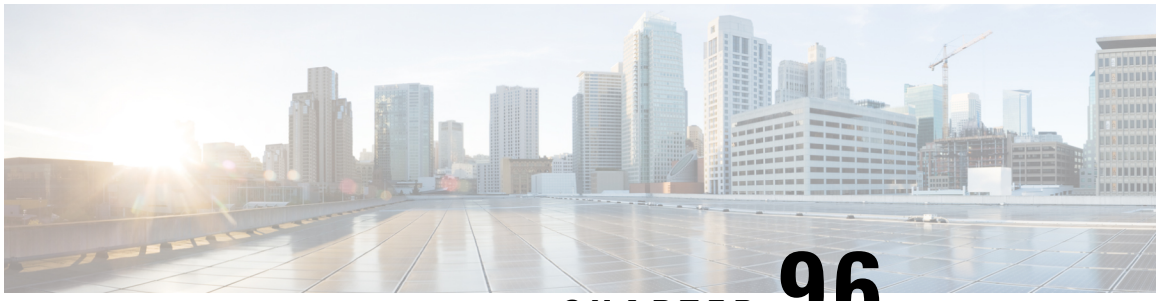
The following is a sample output from the **show vlan access-map** privileged EXEC command:

```
Device# show vlan access-map

Vlan access-map "m1"  10
  Match clauses:
    ipv6 address: ip2
  Action: drop
```

The following is a sample output from the **show ipv6 access-lists** privileged EXEC command. The output shows only IPv6 access lists configured on the switch.

```
Device# show ipv6 access-list

IPv6 access list inbound
permit tcp any any eq bgp (8 matches) sequence 10
```

```
permit tcp any any eq telnet (15 matches) sequence 20
permit udp any any sequence 30
IPv6 access list outbound
deny udp any any sequence 10
deny tcp any any eq telnet sequence 20
```



CHAPTER 96

Object Groups for ACLs

The Object Groups for ACLs feature lets you classify users, devices, or protocols into groups and apply those groups to access control lists (ACLs) to create access control policies for those groups. This feature lets you use object groups instead of individual IP addresses, protocols, and ports, which are used in conventional ACLs. This feature allows multiple access control entries (ACEs), but now you can use each ACE to allow an entire group of users to access a group of servers or services or to deny them from doing so.

In large networks, the number of ACLs can be large (hundreds of lines) and difficult to configure and manage, especially if the ACLs frequently change. Object group-based ACLs are smaller, more readable, and easier to configure and manage than conventional ACLs, simplifying static and dynamic ACL deployments for large user access environments on Cisco IOS routers.

Cisco IOS Firewall benefits from object groups, because they simplify policy creation (for example, group A has access to group A services).

Restrictions for Object Groups for ACLs

- You can use object groups only in extended named and numbered ACLs.
- Object group-based ACLs support only IPv4 or IPv6 addresses.
- Object group-based ACLs support only Layer 3 interfaces (such as routed interfaces and VLAN interfaces) and sub-interfaces.
- Object group-based ACLs are not supported with IPsec.
- ACL statements using object groups will be ignored on packets that are sent to RP for processing.
- The number of object group-based ACEs supported in an ACL varies depending on platform, subject to TCAM availability.

Information About Object Groups for ACLs

You can configure conventional ACEs and ACEs that refer to object groups in the same ACL.

You can use object group-based ACLs with quality of service (QoS) match criteria, Cisco IOS Firewall, Dynamic Host Configuration Protocol (DHCP), and any other features that use extended ACLs. In addition, you can use object group-based ACLs with multicast traffic.

When there are many inbound and outbound packets, using object group-based ACLs increases performance when compared to conventional ACLs. Also, in large configurations, this feature reduces the storage needed in NVRAM, because using object groups in ACEs means that you do not need to define an individual ACE for every address and protocol pairing.

Object Groups

An object group can contain a single object (such as a single IP address, network, or subnet) or multiple objects (such as a combination of multiple IP addresses, networks, or subnets).

A typical access control entry (ACE) allows a group of users to have access only to a specific group of servers. In an object group-based access control list (ACL), you can create a single ACE that uses an object group name instead of creating many ACEs (which requires each ACE to have a different IP address). A similar object group (such as a protocol port group) can be extended to provide access only to a set of applications for a user group. ACEs can have object groups for the source only, destination only, none, or both.

You can use object groups to separate the ownership of the components of an ACE. For example, each department in an organization controls its group membership, and the administrator owns the ACE itself to control which departments can contact one another.

You can use object groups in features that use Cisco Policy Language (CPL) class maps.

This feature supports two types of object groups for grouping ACL parameters: network object groups and service object groups. Use these object groups to group IP addresses, protocols, protocol services (ports), and Internet Control Message Protocol (ICMP) types.

Objects Allowed in Network Object Groups

A network object group is a group of any of the following objects:

- Any IP address—includes a range from 0.0.0.0 to 255.255.255.255 (This is specified using the **any** command.)
- Host IP addresses
- Hostnames
- Other network object groups
- Subnets
- Host IP addresses
- Network address of group members
- Nested object groups

Objects Allowed in Service Object Groups

A service object group is a group of any of the following objects:

- Source and destination protocol ports (such as Telnet or Simple Network Management Protocol [SNMP])
- Internet Control Message Protocol (ICMP) types (such as echo, echo-reply, or host-unreachable)
- Top-level protocols (such as Encapsulating Security Payload [ESP], TCP, or UDP)
- Other service object groups

ACLs Based on Object Groups

All features that use or reference conventional access control lists (ACLs) are compatible with object-group-based ACLs, and the feature interactions for conventional ACLs are the same with object-group-based ACLs. This feature extends the conventional ACLs to support object-group-based ACLs and also adds new keywords and the source and destination addresses and ports.

You can add, delete, or change objects in an object group membership list dynamically (without deleting and redefining the object group). Also, you can add, delete, or change objects in an object group membership list without redefining the ACL access control entry (ACE) that uses the object group. You can add objects to groups, delete them from groups, and then ensure that changes are correctly functioning within the object-group-based ACL without reapplying the ACL to the interface.

You can configure an object-group-based ACL multiple times with a source group only, a destination group only, or both source and destination groups.

You cannot delete an object group that is used within an ACL or a class-based policy language (CPL) policy.

How to Configure Object Groups for ACLs

To configure object groups for ACLs, you first create one or more object groups. These can be any combination of network object groups (groups that contain objects such as, host addresses and network addresses) or service object groups (which use operators such as **lt**, **eq**, **gt**, **neq**, and **range** with port numbers). Then, you create access control entries (ACEs) that apply a policy (such as **permit** or **deny**) to those object groups.

Creating a Network Object Group

A network object group that contains a single object (such as a single IP address, a hostname, another network object group, or a subnet) or multiple objects with a network object-group-based ACL to create access control policies for the objects.

Perform this task to create a network object group.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**object-group network** *object-group-name* **Example:** `Device(config)# object-group network my-network-object-group`	Defines the object group name and enters network object-group configuration mode.
Step 4	**description** *description-text* **Example:** `Device(config-network-group)# description test engineers`	(Optional) Specifies a description of the object group. • You can use up to 200 characters.
Step 5	**host** {*host-address* | *host-name*} **Example:** `Device(config-network-group)# host 209.165.200.237`	(Optional) Specifies the IP address or name of a host. • If you specify a host address, you must use an IPv4 address.
Step 6	*network-address* {*/nn* | *network-mask*} **Example:** `Device(config-network-group)# 209.165.200.225 255.255.255.224`	(Optional) Specifies a subnet object. • You must specify an IPv4 address for the network address. The default network mask is 255.255.255.255.
Step 7	**group-object** *nested-object-group-name* **Example:** `Device(config-network-group)# group-object my-nested-object-group`	(Optional) Specifies a nested (child) object group to be included in the current (parent) object group. • The type of child object group must match that of the parent (for example, if you are creating a network object group, you must specify another network object group as the child). • You can use duplicated objects in an object group only via nesting of group objects. For example, if object 1 is in both group A and group B, you can define a group C that includes both A and B. However, you cannot include a group object that causes the group hierarchy to become circular (for example, you cannot include group A in group B and then also include group B in group A).

	Command or Action	Purpose
		• You can use an unlimited number of levels of nested object groups (however, a maximum of two levels is recommended).
Step 8	Repeat the steps until you have specified objects on which you want to base your object group.	—
Step 9	**end** **Example:** `Device(config-network-group)# end`	Exits network object-group configuration mode and returns to privileged EXEC mode.

Creating a Service Object Group

Use a service object group to specify TCP and/or UDP ports or port ranges. When the service object group is associated with an access control list (ACL), this service object-group-based ACL can control access to ports.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**object-group service** *object-group-name* **Example:** `Device(config)# object-group service my-service-object-group`	Defines an object group name and enters service object-group configuration mode.
Step 4	**description** *description-text* **Example:** `Device(config-service-group)# description test engineers`	(Optional) Specifies a description of the object group. • You can use up to 200 characters.
Step 5	*protocol* **Example:** `Device(config-service-group)# ahp`	(Optional) Specifies an IP protocol number or name.
Step 6	{**tcp** | **udp** | **tcp-udp**} [**source** {{[**eq**] | **lt** | **gt**} *port1* | **range** *port1 port2*}] [{[**eq**] | **lt** | **gt**} *port1* | **range** *port1 port2*]	(Optional) Specifies TCP, UDP, or both.

	Command or Action	Purpose
	Example: `Device(config-service-group)# tcp-udp range 2000 2005`	
Step 7	**icmp** *icmp-type* **Example:** `Device(config-service-group)# icmp conversion-error`	(Optional) Specifies the decimal number or name of an Internet Control Message Protocol (ICMP) type.
Step 8	**group-object** *nested-object-group-name* **Example:** `Device(config-service-group)# group-object my-nested-object-group`	(Optional) Specifies a nested (child) object group to be included in the current (parent) object group. • The type of child object group must match that of the parent (for example, if you are creating a network object group, you must specify another network object group as the child). • You can use duplicated objects in an object group only via nesting of group objects. For example, if object 1 is in both group A and group B, you can define a group C that includes both A and B. However, you cannot include a group object that causes the group hierarchy to become circular (for example, you cannot include group A in group B and then also include group B in group A). • You can use an unlimited number of levels of nested object groups (however, a maximum of two levels is recommended).
Step 9	Repeat the steps to specify the objects on which you want to base your object group.	—
Step 10	**end** **Example:** `Device(config-service-group)# end`	Exits service object-group configuration mode and returns to privileged EXEC mode.

Creating an Object-Group-Based ACL

When creating an object-group-based access control list (ACL), configure an ACL that references one or more object groups. As with conventional ACLs, you can associate the same access policy with one or more interfaces.

You can define multiple access control entries (ACEs) that reference object groups within the same object-group-based ACL. You can also reuse a specific object group in multiple ACEs.

Perform this task to create an object-group-based ACL.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip access-list extended** *access-list-name* **Example:** `Device(config)# ip access-list extended nomarketing`	Defines an extended IP access list using a name and enters extended access-list configuration mode.
Step 4	**remark** *remark* **Example:** `Device(config-ext-nacl)# remark protect server by denying access from the Marketing network`	(Optional) Adds a comment about the configured access list entry. • A remark can precede or follow an access list entry. • In this example, the remark reminds the network administrator that the subsequent entry denies the Marketing network access to the interface.
Step 5	**deny** *protocol source* [*source-wildcard*] *destination* [*destination-wildcard*] [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**established**] [**log** | **log-input**] [**time-range** *time-range-name*] [**fragments**] **Example:** `Device(config-ext-nacl)# deny ip 209.165.200.244 255.255.255.224 host 209.165.200.245 log` `Example based on object-group:` `Router(config)# object-group network my_network_object_group` `Router(config-network-group)# 209.165.200.224 255.255.255.224` `Router(config-network-group)# exit` `Router(config)# object-group network my_other_network_object_group` `Router(config-network-group)# host 209.165.200.245` `Router(config-network-group)# exit` `Router(config)# ip access-list extended`	(Optional) Denies any packet that matches all conditions specified in the statement. • Optionally use the **object-group** *service-object-group-name* keyword and argument as a substitute for the *protocol.* argument • Optionally use the **object-group** *source-network-object-group-name* keyword and argument as a substitute for the *source source-wildcard.* arguments • Optionally use the **object-group** *destination-network-object-group-name* keyword and argument as a substitute for the *destination destination-wildcard.* arguments • If the *source-wildcard* or *destination-wildcard*is omitted, a wildcard mask of 0.0.0.0 is assumed, which matches

	Command or Action	Purpose
	nomarketing Router(config-ext-nacl)# deny ip object-group my_network_object_group object-group my_other_network_object_group log	all bits of the source or destination address, respectively. • Optionally use the **any** keyword as a substitute for the *source source-wildcard* or *destination destination-wildcard* to specify the address and wildcard of 0.0.0.0 255.255.255.255. • Optionally use the **host** *source* keyword and argument to indicate a source and source wildcard of *source* 0.0.0.0 or the **host** *destination* keyword and argument to indicate a destination and destination wildcard of *destination* 0.0.0.0. • In this example, packets from all sources are denied access to the destination network 209.165.200.244. Logging messages about packets permitted or denied by the access list are sent to the facility configured by the **logging facility** command (for example, console, terminal, or syslog). That is, any packet that matches the access list will cause an informational logging message about the packet to be sent to the configured facility. The level of messages logged to the console is controlled by the **logging console** command.
Step 6	**remark** *remark* **Example:** Device(config-ext-nacl)# **remark allow TCP from any source to any destination**	(Optional) Adds a comment about the configured access list entry. • A remark can precede or follow an access list entry.
Step 7	**permit** *protocol source* [*source-wildcard*] *destination* [*destination-wildcard*] [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**established**] [**log** | **log-input**] [**time-range** *time-range-name*] [**fragments**] **Example:** Device(config-ext-nacl)# **permit tcp any any**	Permits any packet that matches all conditions specified in the statement. • Every access list needs at least one permit statement. • Optionally use the **object-group** *service-object-group-name* keyword and argument as a substitute for the *protocol.* • Optionally use the **object-group** *source-network-object-group-name* keyword and argument as a substitute for the *source source-wildcard.*

	Command or Action	Purpose
		• Optionally use the **object-group** *destination-network-object-group-name* keyword and argument as a substitute for the *destination destination-wildcard.* • If *source-wildcard* or *destination-wildcard* is omitted, a wildcard mask of 0.0.0.0 is assumed, which matches on all bits of the source or destination address, respectively. • Optionally use the **any**keyword as a substitute for the *source source-wildcard* or *destination destination-wildcard* to specify the address and wildcard of 0.0.0.0 255.255.255.255. • In this example, TCP packets are allowed from any source to any destination. • Use the **log-input** keyword to include input interface, source MAC address, or virtual circuit in the logging output.
Step 8	Repeat the steps to specify the fields and values on which you want to base your access list.	Remember that all sources not specifically permitted are denied by an implicit **deny** statement at the end of the access list.
Step 9	**end** **Example:** `Device(config-ext-nacl)# end`	Exits extended access-list configuration mode and returns to privileged EXEC mode.

Applying an Object Group-Based ACL to an Interface

Use the **ip access-group** command to apply an object group-based ACL to an interface. An object group-based access control list (ACL) can be used to control traffic on the interface it is applied to.

Perform this task to apply an object group-based ACL to an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**interface** *type number* **Example:** `Device(config)# interface vlan 100`	Specifies the interface and enters interface configuration mode.
Step 4	**ip access-group** {*access-list-name* | *access-list-number*} {**in** | **out**} **Example:** `Device(config-if)# ip access-group my-ogacl-policy in`	Applies the ACL to the interface and specifies whether to filter inbound or outbound packets.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying Object Groups for ACLs

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**show object-group** [*object-group-name*] **Example:** `Device# show object-group my-object-group`	Displays the configuration in the named or numbered object group (or in all object groups if no name is entered).
Step 3	**show ip access-list** [*access-list-name*] **Example:** `Device# show ip access-list my-ogacl-policy`	Displays the contents of the named or numbered access list or object group-based ACL (or for all access lists and object group-based ACLs if no name is entered).

Configuration Examples for Object Groups for ACLs

Example: Creating a Network Object Group

The following example shows how to create a network object group named my-network-object-group, which contains two hosts and a subnet as objects:

```
Device> enable
Device# configure terminal
Device(config)# object-group network my-network-object-group
Device(config-network-group)# description test engineers
Device(config-network-group)# host 209.165.200.237
Device(config-network-group)# host 209.165.200.238

Device(config-network-group)# 209.165.200.241 255.255.255.224
Device(config-network-group)# end
```

The following example shows how to create a network object group named my-company-network, which contains two hosts, a subnet, and an existing object group (child) named my-nested-object-group as objects:

```
Device> enable
Device# configure terminal
Device(config)# object-group network my-company-network
Device(config-network-group)# host host1
Device(config-network-group)# host 209.165.200.242
Device(config-network-group)# 209.165.200.225 255.255.255.224
Device(config-network-group)# group-object my-nested-object-group
Device(config-network-group)# end
```

Example: Creating a Service Object Group

The following example shows how to create a service object group named my-service-object-group, which contains several ICMP, TCP, UDP, and TCP-UDP protocols and an existing object group named my-nested-object-group as objects:

```
Device> enable
Device# configure terminal
Device(config)# object-group service my-service-object-group
Device(config-service-group)# icmp echo
Device(config-service-group)# tcp smtp
Device(config-service-group)# tcp telnet
Device(config-service-group)# tcp source range 1 65535 telnet
Device(config-service-group)# tcp source 2000 ftp
Device(config-service-group)# udp domain
Device(config-service-group)# tcp-udp range 2000 2005
Device(config-service-group)# group-object my-nested-object-group
Device(config-service-group)# end
```

Example: Creating an Object Group-Based ACL

The following example shows how to create an object-group-based ACL that permits packets from the users in my-network-object-group if the protocol ports match the ports specified in my-service-object-group:

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended my-ogacl-policy
Device(config-ext-nacl)# permit object-group my-service-object-group object-group
my-network-object-group any
Device(config-ext-nacl)# deny tcp any any
Device(config-ext-nacl)# end
```

Applying an Object Group-Based ACL to an Interface

Use the **ip access-group** command to apply an object group-based ACL to an interface. An object group-based access control list (ACL) can be used to control traffic on the interface it is applied to.

Perform this task to apply an object group-based ACL to an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface vlan 100`	Specifies the interface and enters interface configuration mode.
Step 4	**ip access-group** {*access-list-name* | *access-list-number*} {**in** | **out**} **Example:** `Device(config-if)# ip access-group my-ogacl-policy in`	Applies the ACL to the interface and specifies whether to filter inbound or outbound packets.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example: Verifying Object Groups for ACLs

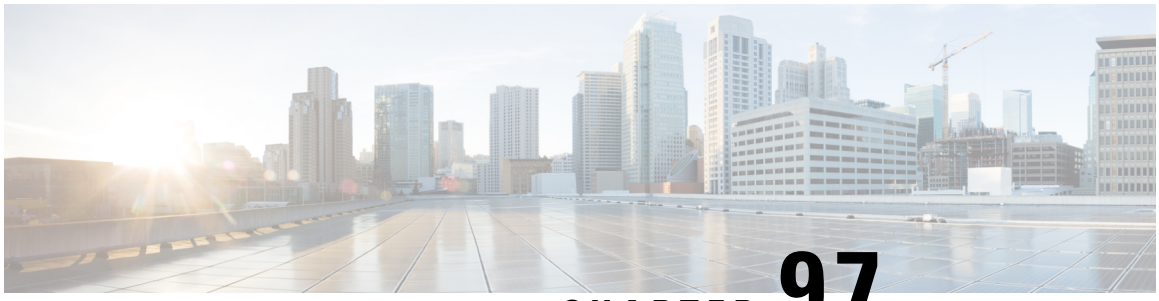
The following example shows how to display all object groups:

```
Device# show object-group

Network object group auth-proxy-acl-deny-dest
 host 209.165.200.235
Service object group auth-proxy-acl-deny-services
 tcp eq www
 tcp eq 443
Network object group auth-proxy-acl-permit-dest
 209.165.200.226 255.255.255.224
 209.165.200.227 255.255.255.224
 209.165.200.228 255.255.255.224
 209.165.200.229 255.255.255.224
 209.165.200.246 255.255.255.224
 209.165.200.230 255.255.255.224
 209.165.200.231 255.255.255.224
 209.165.200.232 255.255.255.224
 209.165.200.233 255.255.255.224
 209.165.200.234 255.255.255.224
Service object group auth-proxy-acl-permit-services
```

```
 tcp eq www
 tcp eq 443
```

The following example shows how to display information about specific object-group-based ACLs:

```
Device# show ip access-list my-ogacl-policy

Extended IP access list my-ogacl-policy
10 permit object-group eng_service any any
```



CHAPTER 97

Configuring IP Source Guard

Information About IP Source Guard

IP Source Guard

You can use IP source guard (IPSG) to prevent traffic attacks if a host tries to use the IP address of its neighbor and you can enable IP source guard when DHCP snooping is enabled on an untrusted interface.

After IPSG is enabled on an interface, the switch blocks all IP traffic received on the interface except for DHCP packets allowed by DHCP snooping.

The switch uses a source IP lookup table in hardware to bind IP addresses to ports. For IP and MAC filtering, a combination of source IP and source MAC lookups are used. IP traffic with a source IP address in the binding table is allowed, all other traffic is denied.

The IP source binding table has bindings that are learned by DHCP snooping or are manually configured (static IP source bindings). An entry in this table has an IP address, its associated MAC address, and its associated VLAN number. The switch uses the IP source binding table only when IP source guard is enabled.

IPSG is supported only on Layer 2 ports, including access and trunk ports. You can configure IPSG with source IP address filtering or with source IP and MAC address filtering.

IP Source Guard for Static Hosts

Note Do not use IPSGfor static hosts on uplink ports or trunk ports.

IPSG for static hosts extends the IPSG capability to non-DHCP and static environments. The previous IPSG used the entries created by DHCP snooping to validate the hosts connected to a switch. Any traffic received from a host without a valid DHCP binding entry is dropped. This security feature restricts IP traffic on nonrouted Layer 2 interfaces. It filters traffic based on the DHCP snooping binding database and on manually

configured IP source bindings. The previous version of IPSG required a DHCP environment for IPSG to work.

IPSG for static hosts allows IPSG to work without DHCP. IPSG for static hosts relies on IP device tracking-table entries to install port ACLs. The switch creates static entries based on ARP requests or other IP packets to maintain the list of valid hosts for a given port. You can also specify the number of hosts allowed to send traffic to a given port. This is equivalent to port security at Layer 3.

IPSG for static hosts also supports dynamic hosts. If a dynamic host receives a DHCP-assigned IP address that is available in the IP DHCP snooping table, the same entry is learned by the IP device tracking table. When you enter the **show device-tracking database** EXEC command, the IP device tracking table displays the entries as ACTIVE.

Note Some IP hosts with multiple network interfaces can inject some invalid packets into a network interface. The invalid packets contain the IP or MAC address for another network interface of the host as the source address. The invalid packets can cause IPSG for static hosts to connect to the host, to learn the invalid IP or MAC address bindings, and to reject the valid bindings. Consult the vender of the corresponding operating system and the network interface to prevent the host from injecting invalid packets.

IPSG for static hosts initially learns IP or MAC bindings dynamically through an ACL-based snooping mechanism. IP or MAC bindings are learned from static hosts by ARP and IP packets. They are stored in the device tracking database. When the number of IP addresses that have been dynamically learned or statically configured on a given port reaches a maximum, the hardware drops any packet with a new IP address. To resolve hosts that have moved or gone away for any reason, IPSG for static hosts leverages IP device tracking to age out dynamically learned IP address bindings. This feature can be used with DHCP snooping. Multiple bindings are established on a port that is connected to both DHCP and static hosts. For example, bindings are stored in both the device tracking database as well as in the DHCP snooping binding database.

IP Source Guard Configuration Guidelines

- You can configure static IP bindings only on nonrouted ports. If you enter the **ip source binding** *mac-address* **vlan** *vlan-id ip-address* **interface** *interface-id* global configuration command on a routed interface, this error message appears:

  ```
  Static IP source binding can only be configured on switch port.
  ```

- When IP source guard with source IP filtering is enabled on an interface, DHCP snooping must be enabled on the access VLAN for that interface.
- If you are enabling IP source guard on a trunk interface with multiple VLANs and DHCP snooping is enabled on all the VLANs, the source IP address filter is applied on all the VLANs.

Note If IP source guard is enabled and you enable or disable DHCP snooping on a VLAN on the trunk interface, the switch might not properly filter traffic.

- You can enable this feature when 802.1x port-based authentication is enabled.

How to Configure IP Source Guard

Enabling IP Source Guard

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be configured, and enters interface configuration mode.
Step 4	**ip verify source** [**mac-check**] **Example:** `Device(config-if)# ip verify source`	Enables IP source guard with source IP address filtering. (Optional) **mac-check**: Enables IP Source Guard with source IP address and MAC address filtering.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 6	**ip source binding** *mac-address* **vlan** *vlan-id* *ip-address* **interface** *interface-id* **Example:** `Device(config)# ip source binding 0100.0230.0002 vlan 11 10.0.0.4 interface gigabitethernet1/1`	Adds a static IP source binding. Enter this command for each static binding.
Step 7	**end** **Example:** `Device(config)# end`	Exits global configuration mode and reutrns to privileged EXEC mode.

Configuring IP Source Guard for Static Hosts on a Layer 2 Access Port

You must configure the **ip device-tracking policy** *policy-name* interface configuration command globally for IPSG for static hosts to work. If you only configure this command on a port without enabling device tracking or by setting an IP device tracking policy number on that interface, IPSG with static hosts rejects all the IP traffic from that interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/1`	Enters interface configuration mode.
Step 4	**device-tracking** **Example:** `Device(config-if)#device-tracking`	Enables device tracking.
Step 5	**switchport mode access** **Example:** `Device(config-if)#switchport mode access`	Configures a port as access.
Step 6	**switchport access vlan** *vlan-id* **Example:** `Device(config-if)#switchport access vlan 10`	Configures the VLAN for this port.
Step 7	**ip verify source**[**tracking**] [**mac-check**] **Example:** `Device(config-if)#ip verify source tracking mac-check`	Enables IP source guard with source IP address filtering. (Optional) **tracking**: Enables IP source guard for static hosts. (Optional) **mac-check**: Enables MAC address filtering. The command **ip verify source tracking mac-check**enables IP source guard for static hosts with MAC address filtering.

	Command or Action	Purpose
Step 8	**ip device-tracking policy** *policy-name* **Example:** `Device(config-if)#ip device-tracking policy p1`	Enables device tracking policy interface.
Step 9	**limit address-count** *limit-number* **Example:** `Device(config-if)#limit address-count 20`	Configures the limit address count.
Step 10	**interface** *interface-id* **Example:** `Device(config)#interface gigabitethernet 1/8`	Enters interface configuration mode.
Step 11	**device-tracking attach-policy** *policy-name* **Example:** `Device(config-if)#device-tracking attach-policy p1`	Enables device tracking and attach policy interface.
Step 12	**end** **Example:** `Device(config-if)#end`	Exits interface configuration mode and returns to privileged EXEC mode.

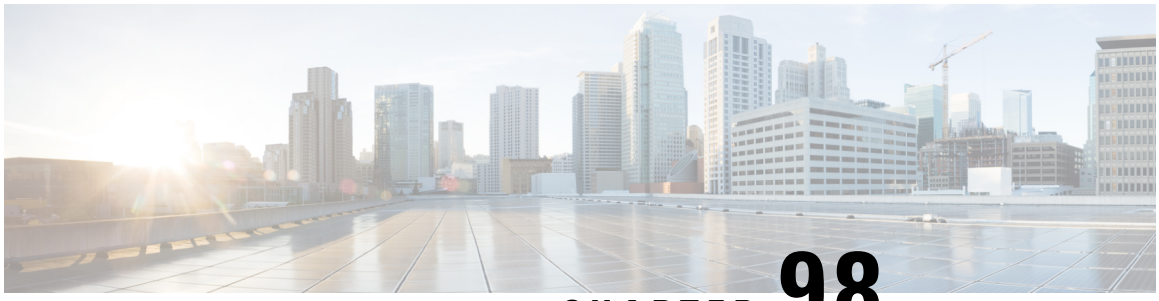
Monitoring IP Source Guard

Table 105: Privileged EXEC show Commands

Command	Purpose
show ip verify source [**interface** *interface-id*]	Displays the IP source guard configuration on the switch or on a specific interface.
show device-tracking database { **address** *network-layer-address* | **details** | **interface** *interface-id* | **mac** *mac-address* | **prefix** *ipv6-prefix* | **vlanid** *vlanid-identifier*}	Displays information about the entries in the IP device tracking table.

Table 106: Interface Configuration Commands

Command	Purpose
ip verify source tracking	Verifies the data source.



CHAPTER 98

Configuring Dynamic ARP Inspection

Restrictions for Dynamic ARP Inspection

This section lists the restrictions and guidelines for configuring Dynamic Address Resolution Protocol (ARP) Inspection on the switch.

- Dynamic ARP inspection is an ingress security feature; it does not perform any egress checking.
- Dynamic ARP inspection is not effective for hosts connected to switches that do not support dynamic ARP inspection or that do not have this feature enabled. Because man-in-the-middle attacks are limited to a single Layer 2 broadcast domain, separate the domain with dynamic ARP inspection checks from the one with no checking. This action secures the ARP caches of hosts in the domain enabled for dynamic ARP inspection.
- Dynamic ARP inspection depends on the entries in the DHCP snooping binding database to verify IP-to-MAC address bindings in incoming ARP requests and ARP responses. Make sure to enable DHCP snooping to permit ARP packets that have dynamically assigned IP addresses.

  When DHCP snooping is disabled or in non-DHCP environments, use ARP ACLs to permit or to deny packets.
- Dynamic ARP inspection is supported on access ports, trunk ports, and EtherChannel ports.

Note Do not enable Dynamic ARP inspection on RSPAN VLANs. If Dynamic ARP inspection is enabled on RSPAN VLANs, Dynamic ARP inspection packets might not reach the RSPAN destination port.

- A physical port can join an EtherChannel port channel only when the trust state of the physical port and the channel port match. Otherwise, the physical port remains suspended in the port channel. A port channel inherits its trust state from the first physical port that joins the channel. Consequently, the trust state of the first physical port need not match the trust state of the channel.

Conversely, when you change the trust state on the port channel, the switch configures a new trust state on all the physical ports that comprise the channel.

- The operating rate for the port channel is cumulative across all the physical ports within the channel. For example, if you configure the port channel with an ARP rate-limit of 400 pps, all the interfaces combined on the channel receive an aggregate 400 pps. The rate of incoming ARP packets on EtherChannel ports is equal to the sum of the incoming rate of packets from all the channel members. Configure the rate limit for EtherChannel ports only after examining the rate of incoming ARP packets on the channel-port members.

  The rate of incoming packets on a physical port is checked against the port-channel configuration rather than the physical-ports configuration. The rate-limit configuration on a port channel is independent of the configuration on its physical ports.

  If the EtherChannel receives more ARP packets than the configured rate, the channel (including all physical ports) is placed in the error-disabled state.

- Make sure to limit the rate of ARP packets on incoming trunk ports. Configure trunk ports with higher rates to reflect their aggregation and to handle packets across multiple dynamic ARP inspection-enabled VLANs. You also can use the **ip arp inspection limit none** interface configuration command to make the rate unlimited. A high rate-limit on one VLAN can cause a denial-of-service attack to other VLANs when the software places the port in the error-disabled state.

- When you enable dynamic ARP inspection on the switch, policers that were configured to police ARP traffic are no longer effective. The result is that all ARP traffic is sent to the CPU.

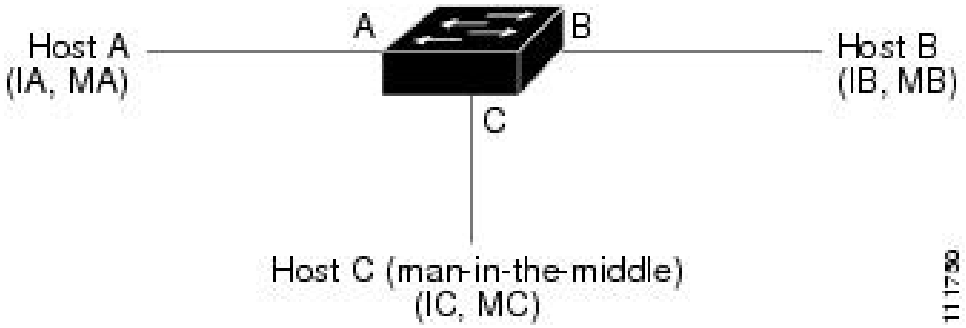
Information About Dynamic ARP Inspection

Understanding Dynamic ARP Inspection

ARP provides IP communication within a Layer 2 broadcast domain by mapping an IP address to a MAC address. For example, Host B wants to send information to Host A but does not have the MAC address of Host A in its ARP cache. Host B generates a broadcast message for all hosts within the broadcast domain to obtain the MAC address associated with the IP address of Host A. All hosts within the broadcast domain receive the ARP request, and Host A responds with its MAC address. However,because ARP allows a gratuitous reply from a host even if an ARP request was not received, an ARP spoofing attack and the poisoning of ARP caches can occur. After the attack, all traffic from the device under attack flows through the attacker's computer and then to the router, switch, or host.

A malicious user can attack hosts, switches, and routers connected to your Layer 2 network by poisoning the ARP caches of systems connected to the subnet and by intercepting traffic intended for other hosts on the subnet. Figure 26-1 shows an example of ARP cache poisoning.

Figure 95: ARP Cache Poisoning



Hosts A, B, and C are connected to the switch on interfaces A, B and C, all of which are on the same subnet. Their IP and MAC addresses are shown in parentheses; for example, Host A uses IP address IA and MAC address MA. When Host A needs to communicate to Host B at the IP layer, it broadcasts an ARP request for the MAC address associated with IP address IB. When the switch and Host B receive the ARP request, they populate their ARP caches with an ARP binding for a host with the IP address IA and a MAC address MA; for example, IP address IA is bound to MAC address MA. When Host B responds, the switch and Host A populate their ARP caches with a binding for a host with the IP address IB and the MAC address MB.

Host C can poison the ARP caches of the switch, Host A, and Host B by broadcasting forged ARP responses with bindings for a host with an IP address of IA (or IB) and a MAC address of MC. Hosts with poisoned ARP caches use the MAC address MC as the destination MAC address for traffic intended for IA or IB. This means that Host C intercepts that traffic. Because Host C knows the true MAC addresses associated with IA and IB, it can forward the intercepted traffic to those hosts by using the correct MAC address as the destination. Host C has inserted itself into the traffic stream from Host A to Host B, the classic *man-in-the middle*attack.

Dynamic ARP inspection is a security feature that validates ARP packets in a network. It intercepts, logs,and discards ARP packets with invalid IP-to-MAC address bindings. This capability protects the network from certain man-in-the-middle attacks.

Dynamic ARP inspection ensures that only valid ARP requests and responses are relayed. The switch performs these activities:

- Intercepts all ARP requests and responses on untrusted ports.
- Verifies that each of these intercepted packets has a valid IP-to-MAC address binding before updating the local ARP cache or before forwarding the packet to the appropriate destination.
- Drops invalid ARP packets.

Dynamic ARP inspection determines the validity of an ARP packet based on valid IP-to-MAC address bindings stored in a trusted database, the DHCP snooping binding database. This database is built by DHCP snooping if DHCP snooping is enabled on the VLANs and on the switch. If the ARP packet is received on a trusted interface, the switch forwards the packet without any checks. On untrusted interfaces, the switch forwards the packet only if it is valid.

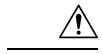
In non-DHCP environments, dynamic ARP inspection can validate ARP packets against user-configured ARP access control lists (ACLs) for hosts with statically configured IP addresses. You define an ARP ACL by using the **arp access-list** *acl-name* global configuration command.

You can configure dynamic ARP inspection to drop ARP packets when the IP addresses in the packets are invalid or when the MAC addresses in the body of the ARP packets do not match the addresses specified in the Ethernet header. Use the **ip arp inspection validate** {[**src-mac**] [**dst-mac**] [**ip**]} global configuration command.

Interface Trust States and Network Security

Dynamic ARP inspection associates a trust state with each interface on the switch. Packets arriving on trusted interfaces bypass all dynamic ARP inspection validation checks, and those arriving on untrusted interfaces undergo the dynamic ARP inspection validation process.

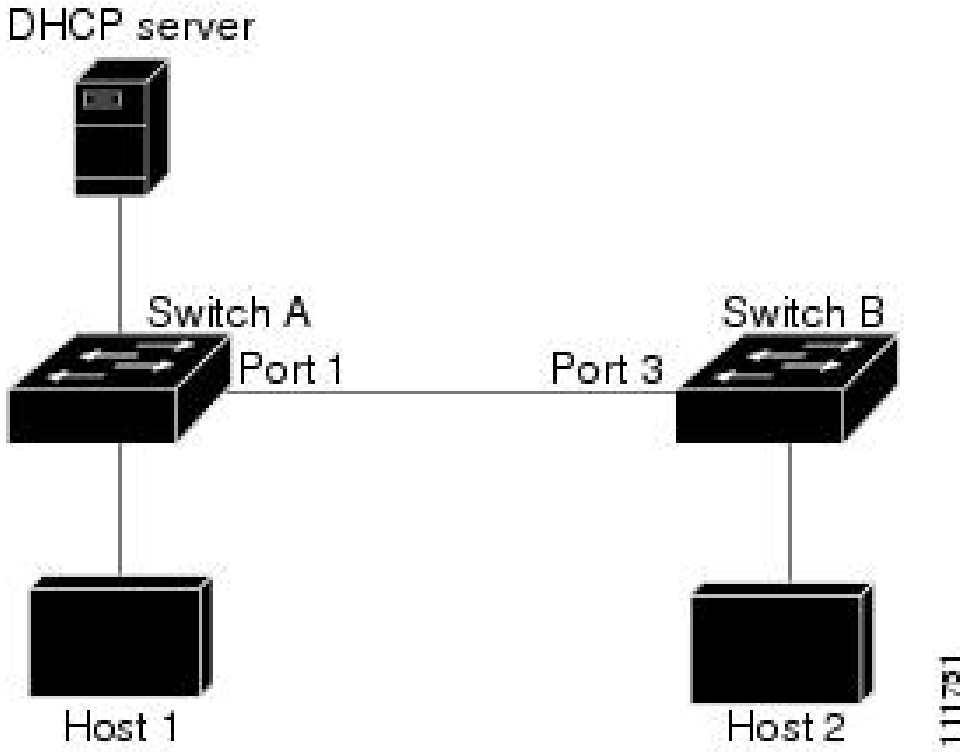
In a typical network configuration, you configure all switch ports connected to host ports as untrusted and configure all switch ports connected to switches as trusted. With this configuration, all ARP packets entering the network from a given switch bypass the security check. No other validation is needed at any other place in the VLAN or in the network. You configure the trust setting by using theip arp inspection trust interface configuration command.



Caution Use the trust state configuration carefully. Configuring interfaces as untrusted when they should betrusted can result in a loss of connectivity.

In the following figure, assume that both Switch A and Switch B are running dynamic ARP inspection on the VLAN that includes Host 1 and Host 2. If Host 1 and Host 2 acquire their IP addresses from the DHCP server connected to Switch A, only Switch A binds the IP-to-MAC address of Host 1. Therefore, if the interface between Switch A and Switch B is untrusted, the ARP packets from Host 1 are dropped by Switch B. Connectivity between Host 1 and Host 2 is lost.

Figure 96: ARP Packet Validation on a VLAN Enabled for Dynamic ARP Inspection



Configuring interfaces to be trusted when they are actually untrusted leaves a security hole in the network. If Switch A is not running dynamic ARP inspection, Host 1 can easily poison the ARP cache of Switch B (and Host 2, if the link between the switches is configured as trusted). This condition can occur even though Switch B is running dynamic ARP inspection.

Dynamic ARP inspection ensures that hosts (on untrusted interfaces) connected to a switch running dynamic ARP inspection do not poison the ARP caches of other hosts in the network. However, dynamic ARP inspection does not prevent hosts in other portions of the network from poisoning the caches of the hosts that are connected to a switch running dynamic ARP inspection.

In cases in which some switches in a VLAN run dynamic ARP inspection and other switches do not, configure the interfaces connecting such switches as untrusted. However, to validate the bindings of packets from nondynamic ARP inspection switches, configure the switch running dynamic ARP inspection with ARP ACLs. When you cannot determine such bindings, at Layer 3, isolate switches running dynamic ARP inspection from switches not running dynamic ARP inspection switches.

Note Depending on the setup of the DHCP server and the network, it might not be possible to validate a given ARP packet on all switches in the VLAN.

Rate Limiting of ARP Packets

The switch CPU performs dynamic ARP inspection validation checks; therefore, the number of incoming ARP packets is rate-limited to prevent a denial-of-service attack. By default, the rate for untrusted interfaces is 15 packets per second (pps). Trusted interfaces are not rate-limited. You can change this setting by using the **ip arp inspection limit** interface configuration command.

When the rate of incoming ARP packets exceeds the configured limit, the switch places the port in the error-disabled state. The port remains in that state until you intervene. You can use the **errdisable recovery** global configuration command to enable error disable recovery so that ports automatically emerge from this state after a specified timeout period.

Relative Priority of ARP ACLs and DHCP Snooping Entries

Dynamic ARP inspection uses the DHCP snooping binding database for the list of valid IP-to-MAC address bindings.

ARP ACLs take precedence over entries in the DHCP snooping binding database. The switch uses ACLs only if you configure them by using the **ip arp inspection filter vlan** global configuration command. The switch first compares ARP packets to user-configured ARP ACLs. If the ARP ACL denies the ARP packet, the switch also denies the packet even if a valid binding exists in the database populated by DHCP snooping.

Logging of Dropped Packets

When the switch drops a packet, it places an entry in the log buffer and then generates system messages on a rate-controlled basis. After the message is generated, the switch clears the entry from the log buffer. Each log entry contains flow information, such as the receiving VLAN, the port number, the source and destination IP addresses, and the source and destination MAC addresses.

You use the **ip arp inspection log-buffer** global configuration command to configure the number of entries in the buffer and the number of entries needed in the specified interval to generate system messages. You specify the type of packets that are logged by using the **ip arp inspection vlan logging** global configuration command.

Default Dynamic ARP Inspection Configuration

Feature	Default Settings
Dynamic ARP inspection	Disabled on all VLANs.
Interface trust state	All interfaces are untrusted.
Rate limit of incoming ARP packets	The rate is 15 pps on untrusted interfaces, assuming that the network is a switched network with a host connecting to as many as 15 new hosts per second. The rate is unlimited on all trusted interfaces. The burst interval is 1 second.
ARP ACLs for non-DHCP environments	No ARP ACLs are defined.
Validation checks	No checks are performed.
Log buffer	When dynamic ARP inspection is enabled, all denied or dropped ARP packets are logged. The number of entries in the log is 32. The number of system messages is limited to 5 per second. The logging-rate interval is 1 second.
Per-VLAN logging	All denied or dropped ARP packets are logged.

Relative Priority of ARP ACLs and DHCP Snooping Entries

Dynamic ARP inspection uses the DHCP snooping binding database for the list of valid IP-to-MAC address bindings.

ARP ACLs take precedence over entries in the DHCP snooping binding database. The switch uses ACLs only if you configure them by using the ip arp inspection filter vlan global configuration command. The switch first compares ARP packets to user-configured ARP ACLs. If the ARP ACL denies the ARP packet, the switch also denies the packet even if a valid binding exists in the database populated by DHCP snooping.

How to Configure Dynamic ARP Inspection

Configuring ARP ACLs for Non-DHCP Environments

This procedure shows how to configure dynamic ARP inspection when Switch B shown in Figure 2 does not support dynamic ARP inspection or DHCP snooping.

If you configure port 1 on Switch A as trusted, a security hole is created because both Switch A and Host 1 could be attacked by either Switch B or Host 2. To prevent this possibility, you must configure port 1 on Switch A as untrusted. To permit ARP packets from Host 2, you must set up an ARP ACL and apply it to VLAN 1. If the IP address of Host 2 is not static (it is impossible to apply the ACL configuration on Switch A) you must separate Switch A from Switch B at Layer 3 and use a router to route packets between them.

Follow these steps to configure an ARP ACL on Switch A. This procedure is required in non-DHCP environments.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**arp access-list** *acl-name* **Example:** `Device(config)# arp access-list arpacl22`	Defines an ARP ACL, and enters ARP access-list configuration mode. By default, no ARP access lists are defined. **Note** At the end of the ARP access list, there is an implicit **deny ip any mac any** command.
Step 4	**permit ip host** *sender-ip* **mac host** *sender-mac* **Example:** `Device(config-arp-nacl))# permit ip host 10.2.2.2 mac host 0018.bad8.3fbd`	Permits ARP packets from the specified host (Host 2). • For*sender-ip*, enter the IP address of Host 2. • For *sender-mac*, enter the MAC address of Host 2.
Step 5	**exit** **Example:** `Device(config-arp-nacl)# exit`	Exits ARP access-list configuration mode and returns to global configuration mode.
Step 6	**ip arp inspection filter** *arp-acl-name* **vlan** *vlan-range* [**static**] **Example:** `Device(config)# ip arp inspection filter arpacl22 vlan 1-2`	Applies ARP ACL to the VLAN. By default, no defined ARP ACLs are applied to any VLAN. • For *arp-acl-name*, specify the name of the ACL created in Step 2. • For *vlan-range*, specify the VLAN that the switches and hosts are in. You can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094.

	Command or Action	Purpose
		• (Optional) Specify **static** to treat implicit denies in the ARP ACL as explicit denies and to drop packets that do not match any previous clauses in the ACL. DHCP bindings are not used. If you do not specify this keyword, it means that there is no explicit deny in the ACL that denies the packet, and DHCP bindings determine whether a packet is permitted or denied if the packet does not match any clauses in the ACL. ARP packets containing only IP-to-MAC address bindings are compared against the ACL. Packets are permitted only if the access list permits them.
Step 7	**interface** *interface-type interface-number* **Example:** `Device(config)# interface gigabitethernt 1/1`	Specifies Switch A interface that is connected to Switch B, and enters interface configuration mode.
Step 8	**no ip arp inspection trust** **Example:** `Device(config-if)# no ip arp inspection trust`	Configures Switch A interface that is connected to Switch B as untrusted. By default, all interfaces are untrusted. For untrusted interfaces, the switch intercepts all ARP requests and responses. It verifies that the intercepted packets have valid IP-to-MAC address bindings before updating the local cache and before forwarding the packet to the appropriate destination. The switch drops invalid packets and logs them in the log buffer according to the logging configuration specified with the **ip arp inspection vlan logging** global configuration command.
Step 9	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 10	**show arp access-list** *acl-name* **Example:** `Device# show arp access-list arpacl22`	Displays information about the named ACLs.
Step 11	**show ip arp inspection vlan** *vlan-range* **Example:** `Device# show ip arp inspection vlan 1-2`	Displays the statistics for the selected range of VLANs.

	Command or Action	Purpose
Step 12	**show ip arp inspection interfaces** **Example:** `Device# show ip arp inspection interfaces`	Displays the trust state and the rate limit of ARP packets for the provided interface.

Configuring Dynamic ARP Inspection in DHCP Environments

Before you begin

This procedure shows how to configure dynamic ARP inspection when two switches support this feature. Host 1 is connected to Switch A, and Host 2 is connected to Switch B. Both switches are running dynamic ARP inspection on VLAN 1 where the hosts are located. A DHCP server is connected to Switch A. Both hosts acquire their IP addresses from the same DHCP server. Therefore, Switch A has the bindings for Host 1 and Host 2, and Switch B has the binding for Host 2.

Note Dynamic ARP inspection depends on the entries in the DHCP snooping binding database to verify IP-to-MAC address bindings in incoming ARP requests and ARP responses. Make sure to enable DHCP snooping to permit ARP packets that have dynamically assigned IP addresses.

Follow these steps to configure dynamic ARP inspection. You must perform this procedure on both switches. This procedure is required.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**show cdp neighbors** **Example:** `Device# show cdp neighbors`	Verify the connection between the switches.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**ip arp inspection vlan** *vlan-range* **Example:** `Device(config)# ip arp inspection vlan 1`	Enable dynamic ARP inspection on a per-VLAN basis. By default, dynamic ARP inspection is disabled on all VLANs. For vlan-range, specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs

	Command or Action	Purpose
		separated by a comma. The range is 1 to 4094. Specify the same VLAN ID for both switches.
Step 5	**Interface***type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface connected to the other switch, and enter interface configuration mode.
Step 6	**ip arp inspection trust** **Example:** `Device(config-if)# ip arp inspection trust`	Configures the connection between the switches as trusted. By default, all interfaces are untrusted. The switch does not check ARP packets that it receives from the other switch on the trusted interface. It simply forwards the packets. For untrusted interfaces, the switch intercepts all ARP requests and responses. It verifies that the intercepted packets have valid IP-to-MAC address bindings before updating the local cache and before forwarding the packet to the appropriate destination. The switch drops invalid packets and logs them in the log buffer according to the logging configuration specified with the ip arp inspection vlan logging global configuration command.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 8	**show ip arp inspection interfaces** **Example:** `Device# show ip arp inspection interfaces`	Verifies the dynamic ARP inspection configuration on interfaces.
Step 9	**show ip arp inspection vlan** *vlan-range* **Example:** `Device# show ip arp inspection vlan 1`	Verifies the dynamic ARP inspection configuration on VLAN.
Step 10	**show ip dhcp snooping binding** **Example:** `Device# show ip dhcp snooping binding`	Verifies the DHCP bindings.
Step 11	**show ip arp inspection statistics vlan** *vlan-range* **Example:**	Checks the dynamic ARP inspection statistics on VLAN.

	Command or Action	Purpose
	`Device# show ip arp inspection statistics vlan 1`	

Limiting the Rate of Incoming ARP Packets

The switch CPU performs dynamic ARP inspection validation checks; therefore, the number of incoming ARP packets is rate-limited to prevent a denial- of-service attack.

When the rate of incoming ARP packets exceeds the configured limit, the switch places the port in the error-disabled state. The port remains in that state until you enable error-disabled recovery so that ports automatically emerge from this state after a specified timeout period.

Note Unless you configure a rate limit on an interface, changing the trust state of the interface also changes its rate limit to the default value for that trust state. After you configure the rate limit, the interface retains the rate limit even when its trust state is changed. If you enter the **no ip arp inspection limit** interface configuration command, the interface reverts to its default rate limit.

Follow these steps to limit the rate of incoming ARP packets. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be rate-limited, and enters interface configuration mode.
Step 4	**ip arp inspection limit {rate pps [burst interval seconds] | none}**	Limits the rate of incoming ARP requests and responses on the interface. The default rate is 15 pps on untrusted interfaces and unlimited on trusted interfaces. The burst interval is 1 second. The keywords have these meanings: • For **rate***pps*, specify an upper limit for the number of incoming packets processed per second. The range is 0 to 2048 pps.

	Command or Action	Purpose
		• (Optional) For **burst interval***seconds*, specify the consecutive interval in seconds, over which the interface is monitored for a high rate of ARP packets. The range is 1 to 15. • For **rate none**, specify no upper limit for the rate of incoming ARP packets that can be processed.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 6	Use the following commands: • **errdisable detect cause arp-inspection** • **errdisable recovery cause arp-inspection** • **errdisable recovery interval** *interval* **Example:** `Device(config)# errdisable recovery cause arp-inspection`	(Optional) Enables error recovery from the dynamic ARP inspection error-disabled state, and configure the dynamic ARP inspection recover mechanism variables. By default, recovery is disabled, and the recovery interval is 300 seconds. For **interval** *interval*, specify the time in seconds to recover from the error-disabled state. The range is 30 to 86400.
Step 7	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Performing Dynamic ARP Inspection Validation Checks

Dynamic ARP inspection intercepts, logs, and discards ARP packets with invalid IP-to-MAC address bindings. You can configure the switch to perform additional checks on the destination MAC address, the sender and target IP addresses, and the source MAC address.

Follow these steps to perform specific checks on incoming ARP packets. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ip arp inspection validate {[src-mac] [dst-mac] [ip]}** **Example:** `Device(config)# ip inspection validate ip`	Performs a specific check on incoming ARP packets. By default, no checks are performed. The keywords have these meanings: • For **src-mac**, check the source MAC address in the Ethernet header against the sender MAC address in the ARP body. This check is performed on both ARP requests and responses. When enabled, packets with different MAC addresses are classified as invalid and are dropped. • For **dst-mac**, check the destination MAC address in the Ethernet header against the target MAC address in ARP body. This check is performed for ARP responses. When enabled, packets with different MAC addresses are classified as invalid and are dropped. • For **ip**, check the ARP body for invalid and unexpected IP addresses. Addresses include 0.0.0.0, 255.255.255.255, and all IP multicast addresses. Sender IP addresses are checked in all ARP requests and responses, and target IP addresses are checked only in ARP responses. You must specify at least one of the keywords. Each command overrides the configuration of the previous command; that is, if a command enables src and dst mac validations, and a second command enables IP validation only, the src and dst mac validations are disabled as a result of the second command.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**show ip arp inspection vlan** *vlan-range* **Example:** `Device# show ip arp insepction vlan 1-2`	Displays the statistics for the selected range of VLANs.

Monitoring DAI

To monitor DAI, use the following commands:

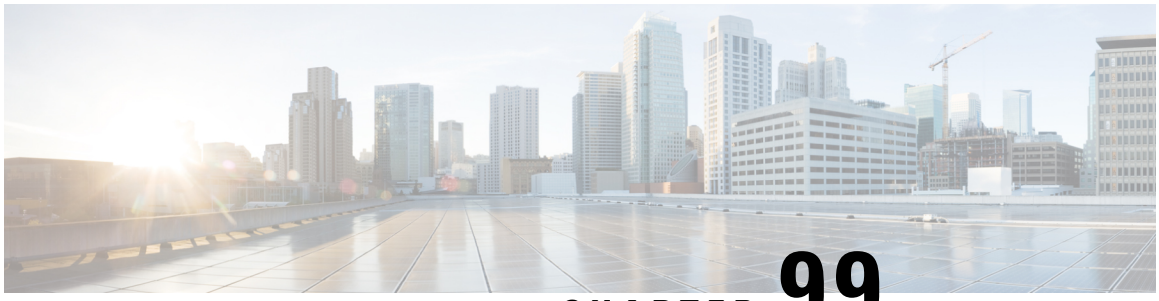
Command	Description
clear ip arp inspection statistics	Clears dynamic ARP inspection statistics.
show ip arp inspection statistics [**vlan** *vlan-range*]	Displays statistics for forwarded, dropped, MAC validation failure, IP validation failure, ACL permitted and denied, and DHCP permitted and denied packets for the specified VLAN. If no VLANs are specified or if a range is specified, displays information only for VLANs with dynamic ARP inspection enabled (active).
clear ip arp inspection log	Clears the dynamic ARP inspection log buffer.
show ip arp inspection log	Displays the configuration and contents of the dynamic ARP inspection log buffer.

For the **show ip arp inspection statistics** command, the switch increments the number of forwarded packets for each ARP request and response packet on a trusted dynamic ARP inspection port. The switch increments the number of ACL or DHCP permitted packets for each packet that is denied by source MAC, destination MAC, or IP validation checks, and the switch increments the appropriate.

Verifying the DAI Configuration

To display and verify the DAI configuration, use the following commands:

Command	Description
show arp access-list [*acl-name*]	Displays detailed information about ARP ACLs.
show ip arp inspection interfaces [interface-id]	Displays the trust state and the rate limit of ARP packets for the specified interface or all interfaces.
show ip arp inspection vlan *vlan-range*	Displays the configuration and the operating state of dynamic ARP inspection for the specified VLAN. If no VLANs are specified or if a range is specified, displays information only for VLANs with dynamic ARP inspection enabled (active).



CHAPTER 99

Configuring Switch Integrated Security Features

Information About SISF

Overview

Switch Integrated Security Features (SISF) is a framework developed to optimize security in Layer 2 domains. It merges the IP Device Tracking (IPDT) and *certain* IPv6 first-hop security (FHS) functionality[8], to simplify the migration from IPv4 to IPv6 stack or a dual-stack.

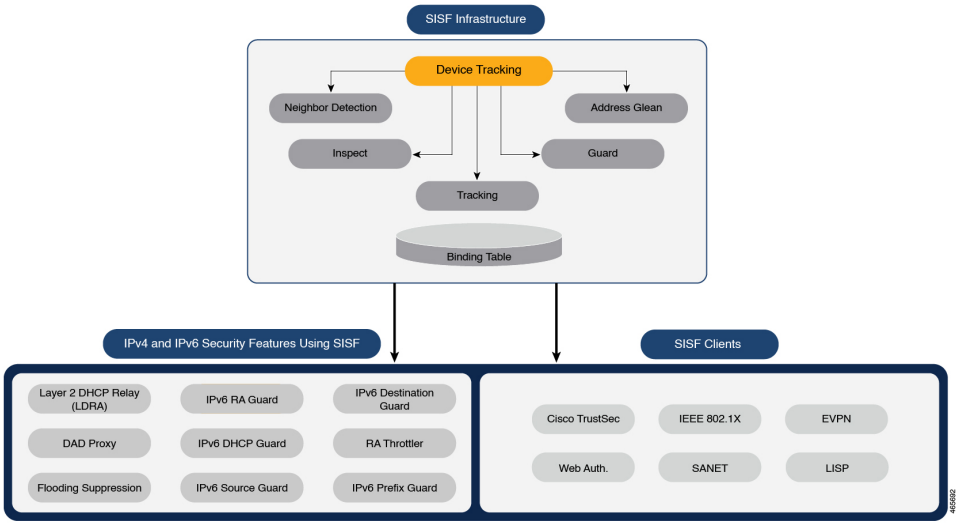
The SISF infrastructure provides a unified database that is used by:

- IPv6 FHS features: IPv6 Router Advertisement (RA) Guard, IPv6 DHCP Guard, Layer 2 DHCP Relay, IPv6 Duplicate Address Detection (DAD) Proxy, Flooding Suppression, IPv6 Source Guard, IPv6 Destination Guard, RA Throttler, and IPv6 Prefix Guard.
- Features like Cisco TrustSec, IEEE 802.1X, Locator ID Separation Protocol (LISP), Ethernet VPN (EVPN), and Web Authentication, which act as clients for SISF.

The following figure illustrates this:

[8] IPv6 Snooping Policy, IPv6 FHS Binding Table Content, and IPv6 Neighbor Discovery Inspection

Figure 97: SISF Framework



Note The terms “SISF” “device-tracking” and “SISF-based device-tracking” are used interchangeably in this document and refer to the same feature. Neither term is used to mean or should be confused with the legacy IPDT or IPv6 Snooping features.

Understanding the SISF Infrastructure

This section explains the various elements of the SISF infrastructure as shown in the SISF Framework above.

The Binding Table

The SISF infrastructure is built around the binding table. The binding table contains information about the hosts that are connected to the ports of a switch and the IP and MAC address of these hosts. This helps to create a physical map of all the hosts that are connected to a switch.

Each entry in a binding table provides the following information about a connected host:

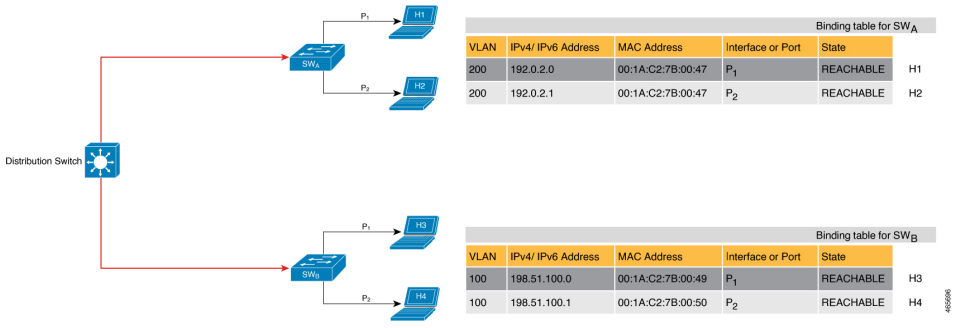
- IPv4 or IPv6 address of the host.
- MAC address of the host. The same MAC address may be linked to an IPv4 and IPv6 address.
- The interface or port on the switch that the host is connected to, and the associated VLAN.
- The state of the entry, which indicates the reachability of the entry.

The following figure shows a simple network topology and a representative binding table for each access switch in the network. SW_A and SW_B are the two access switches in the network. The two access switches are connected to the same distribution switch. H1, H2, H3, H4 are the hosts.

This is an example of a distributed binding table, that is, each access switch in the network has its own table. An alternative set-up could be one centralised binding table on the distribution switch with the entries of SW_A *and* SW_B.

Having a distributed or a centralised binding table is a key design choice in the process of implementing SISF in your network and is covered in greater detail in the Understanding Policy Parameters, on page 1393 section in this chapter.

Figure 98: Binding Table



States and Lifetime of a Binding Table Entry

The state of an entry indicates if the host is reachable or not. The stable states of a binding table entry are: REACHABLE, DOWN, and STALE. When changing from one state to another, an entry may have other temporary or transitional states such as: VERIFY, INCOMPLETE, and TENTATIVE.

How long an entry remains in a given state is determined by its lifetime and by whether or not the entry is validated successfully. The lifetime of an entry can be policy-driven or configured globally.

To configure the REACHABLE, DOWN, and STALE lifetimes, enter the following command in global configuration mode:

device-tracking binding { **reachable-lifetime** { *seconds* | **infinite** } | **stale-lifetime** { *seconds* | **infinite** } | **down-lifetime** { *seconds* | **infinite** } }

State: Reachable

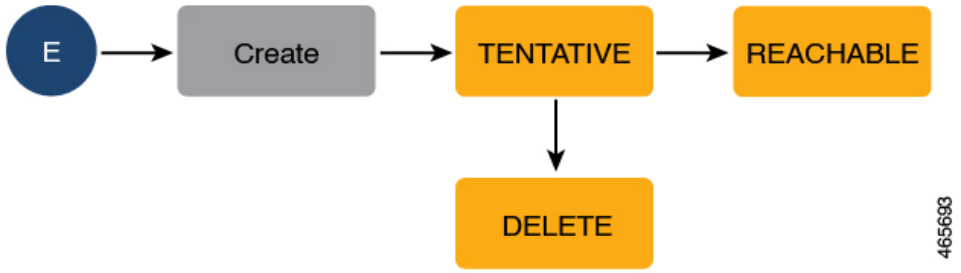
If an entry has this state, it means the host (IP and MAC address) from which a control packet was received, is a verified and valid host. A reachable entry has a default lifetime of 5 minutes. You can also configure a duration. By configuring a reachable-lifetime, you specify how long a host can remain in a REACHABLE state, after the last incoming control packet from that host.

If an event is detected before the entry's reachable lifetime expires, then the reachable lifetime is reset.

To qualify for the REACHABLE state, a new entry goes through the process illustrated in the figure below. The switch detects an event (E), such as an incoming control packet from a connected host and creates an entry. Various events cause the creation of an entry, and these are described in the Binding Table Sources section. The creation of an entry is followed by different transient states, such as TENTATIVE or INCOMPLETE. While in a transitional state, the switch validates and confirms the integrity of the binding entry. If the entry is found to be valid, then the state changes to REACHABLE.

But if an address theft or similar event is detected, then the entry is regarded as invalid and is deleted. For example, if an attacker sends unsolicited neighbor advertisement messages with the same IP as the target IP and its (attacker's) own MAC address to redirect traffic.

Figure 99: Creation of a Reachable Entry

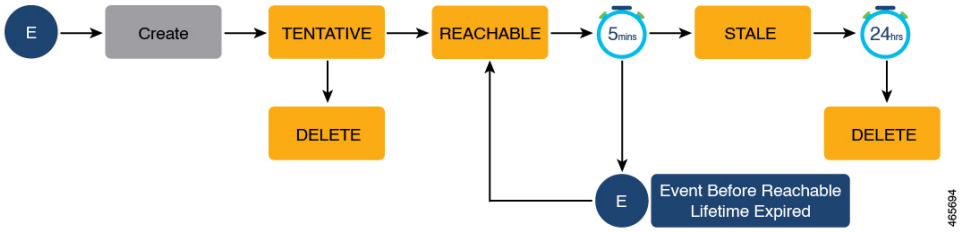


State: Stale

If an entry is in this state it means that the entry's reachable lifetime has expired and the corresponding host is still silent (no incoming packets from the host). A stale entry has a default lifetime of 24 hours. You can also configure a duration. An entry that remains in the STALE state beyond the stale lifetime, is deleted.

This is illustrated in the figure below which depicts the lifecycle of an entry.

Figure 100: Lifecycle of an Entry



State: Down

If an entry is in this state, it means that the host's connecting interface is down. A down entry has a default lifetime of 24 hours. You can also configure a duration. An entry that remains in the DOWN state beyond the down lifetime, is deleted.

Polling a Host and Updating the Binding Table Entry

Polling is a periodic and conditional checking of the host to see the state it is in, whether it is still connected, and whether it is communicating. In addition to determining an entry's state, you can use polling to reconfirm an entry's state.

You can enable polling with the **device-tracking tracking** command in global configuration mode. After you do, you still have the flexibility to turn polling on or off for a particular interface or VLAN. For this, configure the **tracking enable** or **tracking disable** keywords in the policy (the device-tracking configuration mode). When polling is enabled, the switch polls the host at the specified interval, thus reconfirming its reachability for the duration of its reachable lifetime.

When polling is enabled, the switch sends up to three polling requests, after the reachable lifetime expires, at system-determined intervals. You can also configure this interval with the **device-tracking tracking retry-interval** *seconds* command in global configuration mode.

The figure below depicts the lifecycle of an entry where the host is polled. Default reachable and stale lifetimes, and retry intervals are used in figure:

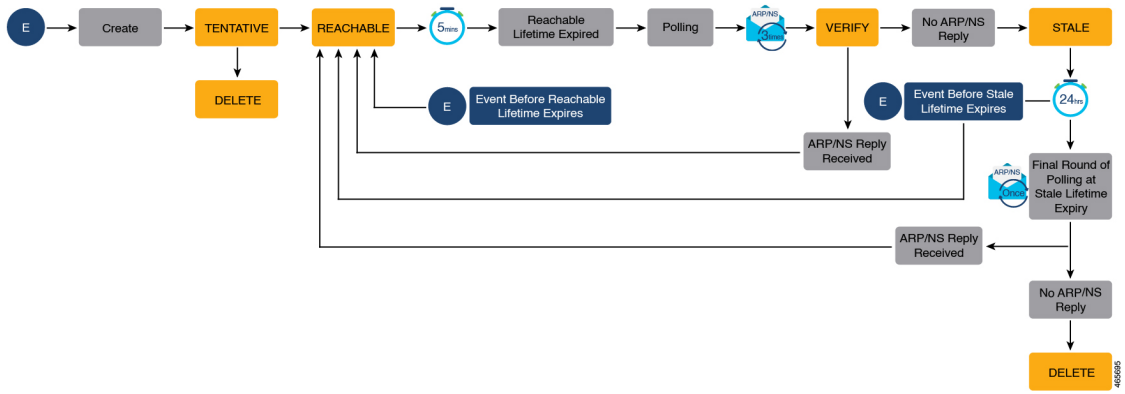
An event (E) is detected and a REACHABLE entry is created.

If an event is detected *during* the reachable lifetime, the reachable lifetime timer is reset.

The switch sends a polling request after the reachable lifetime expires. The switch polls the host up to three times at fixed, system-determined intervals. The polling request may be in the form of a unicast Address Resolution Protocol (ARP) probe or a Neighbor Solicitation message. During this time the state of the entry changes to VERIFY. If a polling response is received (thus confirming reachability of the host), the state of the entry changes back to REACHABLE.

If the switch does not receive a polling response after three attempts, the entry changes to the STALE state. It remains in this state for 24 hours. If an event is detected during the stale lifetime, the state of the entry is changed back to REACHABLE. At expiry of the stale lifetime, the device sends one final polling to ascertain reachability. If this final polling attempt receives a reply, the state of the entry is changed back to REACHABLE. If the final polling attempt does not receive a response, the entry is deleted.

Figure 101: Lifecycle of an Entry Where the Host is Polled



Binding Table Sources

The following are the sources of information and events that cause the creation and update of a binding table entry:

- Learning events that dynamically populate the binding table:
  - Dynamic Host Configuration Protocol (DHCP) negotiation (DHCP REQUEST, and DHCP REPLY). This includes DHCPv4 and DHCPv6.
  - Address Resolution Protocol (ARP) packets.

    ARP packets are throttled to mitigate high CPU utilization scenarios. In a five second window, a maximum of 50 ARP broadcast packets per binding entry are processed by SISF. When the limit is reached, incoming ARP packets are dropped. Note that the limit of 50 in five seconds is for each binding entry, that is, for each source IP. This limit is increased to a maximum of 100 ARP broadcast packets for each source IP. When the limit is reached, incoming ARP packets are dropped.
  - Neighbor Discovery Protocol (NDP) packets.
  - Multiple Identity Association-Nontemporary Address (IA_NA) and Identity Association-Prefix Delegation (IA_PD).

In some cases, a network device can request and receive more than one IPv6 address from the DHCP server. This may be done to provide addresses to multiple clients of the device, such as when a residential gateway requests addresses to distribute to its LAN clients. When the device sends out a DHCPv6 packet, the packet includes all of the addresses that have been assigned to the device.

When SISF analyzes a DHCPv6 packet, it examines the IA_NA (Identity Association-Nontemporary Address) and IA_PD (Identity Association-Prefix Delegation) components of the packet and extracts each IPv6 address contained in the packet. SISF adds each extracted address to the binding table.

Entries created through learning events like the ones listed above are called "dynamic entries". In the output of the **show device-tracking database details** privileged EXEC command, such entries are prefixed with an abbreviation that clarifies the kind of dynamic learning event it was. For example, ARP for ARP packets, ND for NDP packets, and so on.

- Configuring static binding entries.

  If there are silent but reachable hosts in the Layer 2 domain, you can create static binding entries to retain binding information even if the host becomes silent.

  A static binding entry is a binding entry that is manually added to the binding table, by configuring the following command in global configuration mode:

  device-tracking binding vlan *vlan_id* { *ipv4_add ipv6_add ipv6_prefix* } [**interface** *inteface_type_no*] [*48-bit-hardware-address*] [**reachable-lifetime** { *seconds* | **default** | **infinite** } | **tracking** { **default** | **disable** | **enable** [**retry-interval** { *seconds* | **default** }] } [**reachable-lifetime** { *seconds* | **default** | **infinite** }]]

  In the output of the **show device-tracking database details** privileged EXEC command, static entries are prefixed with the letter "S".

  You can configure a reachable lifetime for a static entry. The stale and down lifetime timer is fixed by the system as **infinite** (For an entry in the STALE or DOWN state, the output of the **show device-tracking database** command displays the `Time Left` column as “N/A”). This means that when a static entry enters the STALE or DOWN state it remains in this state, and in the binding table, indefinitely.

  A static entry can be removed from the binding table only by the actions listed below. It cannot be deleted from the binding table by using **clear** commands or by any other event:

  - You remove the entry by configuring the **no** form of the above command.
  - A local entry replaces the static entry.

    A local entry is an entry that is automatically created by the system when you configure an SVI on the device. When configuring the SVI, if you use the same IP address as the static entry then the static entry is replaced with the local entry, because the local entry has a higher priority.

    This replacement of a static entry by a local entry is introduced.

    In the output of the **show device-tracking database details** privileged EXEC command, local entries are prefixed with the letter "L".

  For more information about static binding entries, see the **device-tracking binding** command in the command reference.

Note In addition to the primary or key events listed above, there is a specific scenario in which a ping can result in a device-tracking entry. If a sender's ARP cache or IPv6 neighbor table doesn't have the target's IP address yet, then a ping triggers an ARP packet for IPv4, or ND packet for IPv6. This can result in a device-tracking entry.

But if the target IP is already in the ARP cache or IPv6 neighbour table, no ARP or ND packet is generated when you ping - in which case SISF cannot learn the IP address.

Device-Tracking

SISF-based device-tracking is disabled by default. You can enable the feature on an interface or VLAN.

When you enable the feature, the binding table is created, followed by subsequent maintenance of the binding table.

The events listed in the Binding Table Sources, on page 1391 section act as triggers for SISF-based device-tracking, to track the presence, location, and movement of hosts in the network, to populate and maintain the binding table. For example, if information about a host is learnt by means of an ARP or ND packet, every subsequent ARP or ND packet from the same host acts as an alert for SISF-based device-tracking, to refresh the entry in the binding table, thus indicating if the host is still present in the same location or has moved.

The continuous process of snooping of packets that the switch receives, extraction of device identity (MAC and IP address), and storage of information in the binding table of the switch, ensures binding integrity and maintains the reachability status of the hosts in the binding table.

For information how to enable SISF-based device-tracking, see How to Configure SISF, on page 1403.

Device-Tracking Policy

A device-tracking policy is a set of rules that SISF-based device-tracking follows. The policy dictates which events will be listened to, whether a host will be probed, the wait time before the host is probed, and so on. These rules are referred to as policy parameters.

Note The policy must be attached to an interface or VLAN. Only then is the binding table for that interface or VLAN populated - in accordance with policy parameters.

For information about the various ways in which you can create a policy, see How to Configure SISF, on page 1403.

To display a policy's settings, use the **show device-tracking policy** *policy_name* command in privileged EXEC mode.

Understanding Policy Parameters

Policy parameters are the keywords available for configuration in the device-tracking configuration mode. Each policy parameter addresses one or more aspects of network security.

This section explains the purpose of *some* of the important policy parameters so you can configure your policy to better suit your requirements.

```
Device(config)# device-tracking policy example_policy
Device(config-device-tracking)# ?
device-tracking policy configuration mode:

  device-role           Sets the role of the device attached to the port
  limit                 Specifies a limit
  security-level        setup security level
  tracking              Override default tracking behavior
  trusted-port          setup trusted port
```

For information about all the parameters displayed in the device-tracking configuration mode, see the command reference document of the corresponding release.

Glean versus Guard versus Inspect

When a packet enters the network, SISF extracts the IP and MAC address (the source of the packet) and subsequent action, is dictated by the security-level that is configured in the policy.

Glean, guard, and inspect are the options available under the security-level parameter. Glean is the least secure option, inspect, is moderately secure, and guard, is the most secure.

To configure this parameter in a policy, enter the **security-level** keyword in the device-tracking configuration mode.

Glean

When the security-level is set to **glean**, SISF extracts the IP and MAC address and enters them into the binding table, without any verification. This option therefore does not ensure binding integrity. It may for example, be suited to a set-up where client applications such as IEEE 802.1X or SANET want to only learn about the host and not rely on SISF for authentication.

The only factor that affects the addition of the binding entry for this security-level, is the address count limit. There are separate limits for the maximum number of IPs per port, IPv4 per MAC, and IPv6 per MAC. Entries are rejected once a limit is reached. For more information about this parameter, see Address Count Limits, on page 1401.

Guard

This is the default value for the security-level parameter.

When the security-level is set to **guard**, SISF extracts and verifies the IP and MAC address of packets entering the network. The outcome of the verification determines if a binding entry is added, or updated, or if the packet is dropped and the client is rejected.

The process of verification starts with the search for a matching entry in the database. The database may be centralised or distributed. If a matching entry is not found, a new entry is added.

If a matching entry is found and the points of attachment (MAC, VLAN, or interface) are found to be the same, only the timestamp is updated. If not, the scope of verification is extended to include validation of address ownership. This may include host polling to determine if the change in the point of attachment (a different MAC, or VLAN) is valid. If the change is valid the entry is updated, or if it is a case of theft, the entry is not added to the binding table.

If a binding entry is added or updated, the corresponding client is granted access to the network. If an entry does not pass verification, the corresponding client is rejected.

Note The verification process affects the binding entry and the corresponding incoming packet.

The **guard** security-level supports the *prevention* of IPv4 spoofing. Detection and reporting of IPv4 spoofing is supported since the introductory release of SISF. Further, detection, reporting, and prevention of *IPv6 spoofing* is supported since the introductory release of SISF. For more information, see: Example: Detecting and Preventing Spoofing, on page 1418 .

Inspect

Even though security-level **inspect** is available on the CLI, we recommend not using it. The **glean** and **guard** options described above address most use cases and network requirements.

The security level parameter affects how ARP and ND packets are handled.

Trusted-Port and Device-Role Switch

The **device-role switch** and **trusted-port** options help you design an efficient and scalable secure zone. When used together, these two parameters help you achieve an efficient distribution of the creation of entries in the binding table. This keeps the binding tables size under control.

The **trusted-port** option: Disables the guard function on configured targets. Bindings learned through a trusted-port have preference over bindings learned through any other port. A trusted port is also given preference in case of a collision while making an entry in the table.

The **device-role** option: Indicates the type of device that is facing the port and this can be a node or a switch. To allow the creation of binding entries for a port, you configure the device as a node. To stop the creation of binding entries, you configure the device as switch.

Configuring the device as a switch is suited to multi-switch set-ups, where the possibility of large device tracking tables is very high. Here, a port facing a device (an uplink trunk port) can be configured to stop creating binding entries, and the traffic arriving at such a port can be trusted, because the switch on the other side of the trunk port will have device-tracking enabled and that will have checked the validity of the binding entry.

Note While there are scenarios where configuring only either one of these options may be suitable, the more common use case is for both the **trusted-port** and **device-role switch** options to be configured on the port - the examples below explain this in detail. Possible scenarios where only either one of these options is suited or required have also been described, at the end of this section.

To configure these parameters in a policy, enter the **trusted-port** and **device-role** keywords in the device-tracking configuration mode.

Example: Using Trusted-Port and Device-Role Switch Options in a Multi-Switch Set-Up

The following example explains how the **device-role switch** and **trusted-port** options help to design an efficient and scalable “secure zone”.

In figure "*Multi-Switch Set-Ups Without Trusted-Port and Device-Role Switch Options*" below, SW_A, SW_B, and SW_C are three access switches. They are all connected to a common distribution switch. The only required configuration on the distribution switch in this scenario is to ensure that traffic of any kind is *not* blocked.

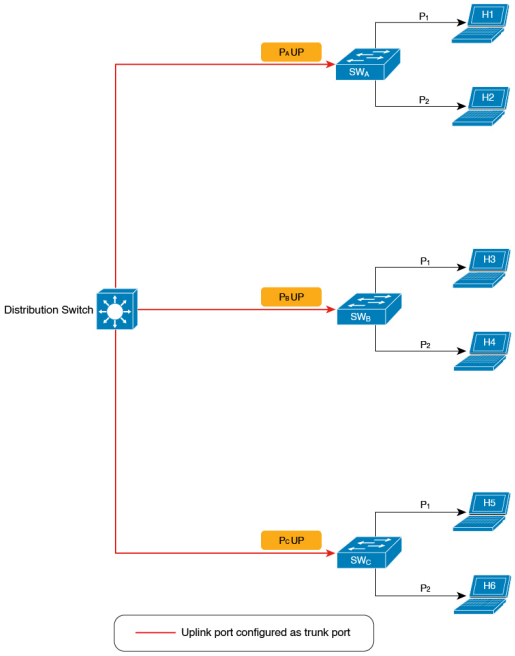
H1, H2, …H6 are the hosts. Each switch has two directly connected hosts. All hosts are communicating with each other, that is, control packets are being transmitted. All hosts are also within the same VLAN boundary. Each switch is receiving control packets from hosts that are directly connected to it, and also from hosts that are connected to other switches. This means SW_A is receiving control packets from H1, H2, …H6 similarly with SW_B and SW_C.

For each switch, the entries of directly connected hosts have interface or port P_1 and P_2 in the binding table. Entries originating from hosts that are connected to other switches have interface or port name P_xUP, to show that they have been learned through the uplink port (x represents the corresponding uplink port for each switch. For example, the entries that SW_A has learnt through its uplink port have interface or port name P_AUP and for SW_B it is P_BUP, and so forth.

The end result is that each switch learns and creates binding entries for all hosts in the set-up.

This scenario displays an inefficient use of the binding table, because each host is being validated multiple times, which does not make it more secure than if just one switch validates host. Secondly, entries for the same host in multiple binding tables could mean that the address count limit is reached sooner. After the limit is reached, any further entries are rejected and required entries may be missed this way.

Figure 102: Multi-Switch Set-Ups Without Trusted-Port and Device-Role Switch Options



Binding table for SW_A

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
100	192.0.2.1	00:1A:C2:7B:00:47	P_1	REACHABLE	H1
100	192.0.2.2	00:1A:C2:7B:00:47	P_2	REACHABLE	H2
200	192.0.2.3	00:1A:C2:7B:00:48	P_AUP	REACHABLE	H3
200	192.0.2.4	00:1A:C2:7B:00:48	P_AUP	REACHABLE	H4
300	192.0.2.5	00:1A:C2:7B:00:49	P_AUP	REACHABLE	H5
300	192.0.2.6	00:1A:C2:7B:00:49	P_AUP	REACHABLE	H6

Binding table for SW_B

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
200	192.0.2.3	00:1A:C2:7B:00:48	P_1	REACHABLE	H3
200	192.0.2.4	00:1A:C2:7B:00:48	P_2	REACHABLE	H4
100	192.0.2.1	00:1A:C2:7B:00:47	P_BUP	REACHABLE	H1
100	192.0.2.2	00:1A:C2:7B:00:47	P_BUP	REACHABLE	H2
300	192.0.2.5	00:1A:C2:7B:00:49	P_BUP	REACHABLE	H5
300	192.0.2.6	00:1A:C2:7B:00:49	P_BUP	REACHABLE	H6

Binding table for SW_C

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
300	192.0.2.5	00:1A:C2:7B:00:49	P_1	REACHABLE	H5
300	192.0.2.6	00:1A:C2:7B:00:49	P_2	REACHABLE	H6
100	192.0.2.1	00:1A:C2:7B:00:47	P_CUP	REACHABLE	H1
100	192.0.2.2	00:1A:C2:7B:00:47	P_CUP	REACHABLE	H2
200	192.0.2.3	00:1A:C2:7B:00:48	P_CUP	REACHABLE	H3
200	192.0.2.4	00:1A:C2:7B:00:48	P_CUP	REACHABLE	H4

465697

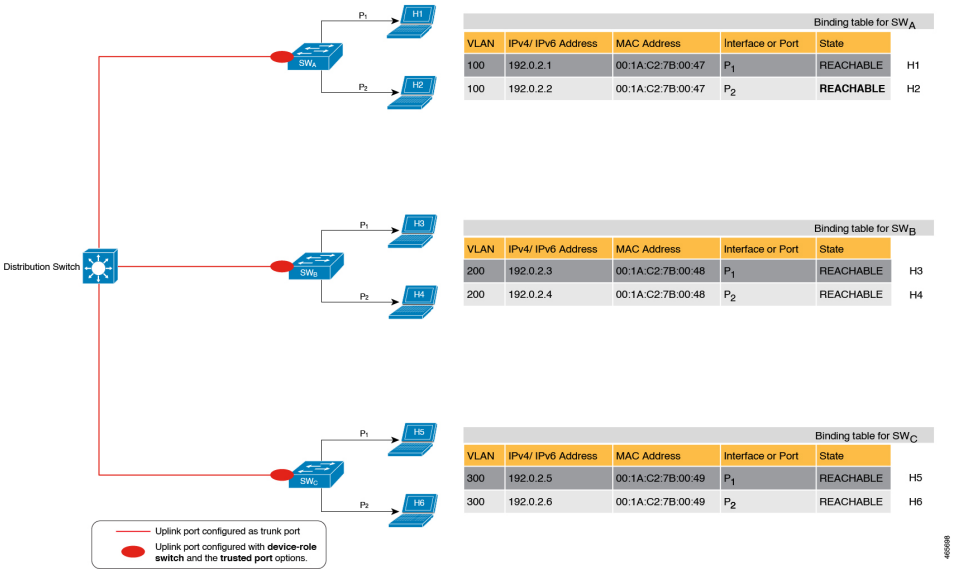
By contrast, see figure "Multi-Switch Set-Ups With Trusted-Port and Device-Role Switch Options" below. Here, when SW_A intercepts the packet of a host that is not attached to it (say H3 which is directly attached to SW_B), it does not create an entry because it detects that H3 is attached to a device that is configured as a switch (**device-role switch** option) and the uplink port of the switch (where the packet came from) is a trusted port (**trusted-port** option).

By creating binding entries only on switches where the host appears on an access port (port P_1 and P_2 of each switch), and not creating entries for a host that appears over an uplink port or trusted port (P_x UP), each switch

in the set-up validates and makes only the required entries, thus achieving an efficient distribution of the creation of binding table entries.

A second advantage of configuring **device-role switch** and **trusted-port** options in a multi-switch scenario is that it prevents duplicate entries when a host, say H1 moves from one switch to another. H1's IP and MAC binding in the earlier location (let's say SW_A) continues to remain there until it reaches the STALE state. But if H1 moves and connects to a second switch, say SW_C, then SW_A receives a duplicate binding entry through the uplink port. In such a situation, if the uplink port of the second switch (SW_C) is configured as a trusted port, SW_A deletes its stale entry. Further, it doesn't create another new binding entry because the SW_C will already have the latest entry and this entry is trusted.

Figure 103: Multi-Switch Set-Ups With Trusted-Port and Device-Role Switch Options



Binding table for SW_A

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
100	192.0.2.1	00:1A:C2:7B:00:47	P_1	REACHABLE	H1
100	192.0.2.2	00:1A:C2:7B:00:47	P_2	**REACHABLE**	H2

Binding table for SW_B

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
200	192.0.2.3	00:1A:C2:7B:00:48	P_1	REACHABLE	H3
200	192.0.2.4	00:1A:C2:7B:00:48	P_2	REACHABLE	H4

Binding table for SW_C

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
300	192.0.2.5	00:1A:C2:7B:00:49	P_1	REACHABLE	H5
300	192.0.2.6	00:1A:C2:7B:00:49	P_2	REACHABLE	H6

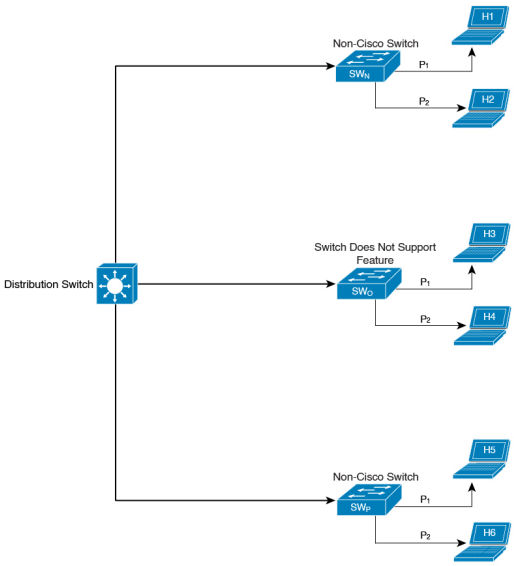
Example: When Not to Use Trusted-Port and Device-Role Switch Options

While the previous example clarifies how a multi-switch set-up with distributed binding tables stands to benefit from the **device-role switch** and **trusted-port** options, it may not suit networks of the following kinds:

- Networks where non-Cisco switches are being used
- Networks where the switch does not support the SISF-based device-tracking feature.

In both cases, we recommended that you not configure the **device-role switch** and **trusted-port** options. Further, we recommended that you maintain a centralised binding table - on the distribution switch. When you do, all the binding entries for all the hosts connected to non-Cisco switches and switches that do not support the feature, are validated by the distribution switch and still secure your network. The figure below illustrates the same.

Figure 104: Centralised Binding Table



Binding table for Distribution Switch

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
100	192.0.2.1	00:1A:C2:7B:00:47	P_NUP	REACHABLE	H1
100	192.0.2.2	00:1A:C2:7B:00:47	P_NUP	REACHABLE	H2
200	192.0.2.3	00:1A:C2:7B:00:48	P_OUP	REACHABLE	H3
200	192.0.2.4	00:1A:C2:7B:00:48	P_OUP	REACHABLE	H4
300	192.0.2.5	00:1A:C2:7B:00:49	P_PUP	REACHABLE	H5
300	192.0.2.6	00:1A:C2:7B:00:49	P_PUP	REACHABLE	H6

465699

Creating an Efficient and Scalable Secure Zone

By using the **trusted-port** and **device-role switch** options in suitable networks and leaving them out in others, you can achieve an efficient and scalable secure zone.

Secure Zones 1, 2 and 3, display three different set-ups and the secure zone that is established in each case.

Secure Zone:	**Secure Zone 1 - Inefficient and Unscalable Secure Zone**	**Secure Zone 2 - Efficient and Scalable Secure Zone When Binding Tables are Decentralized**	**Secure Zone 3: Efficient Secure Zone When Binding Table is Centralized**
Scalability:	Unscalable; each switch has entries of all the hosts in the network	Scalable; each switch as entries of only directly connected hosts	Unscalable; the distribution switch has entries of all hosts in the network
Polling and its effect on the network: n = number of hosts m = number of switches total number of polling requests: = n X m	18 polling requests are being sent (6 hosts x 3 switches). Each host is polled by all the switches in the network (in the absence of the **trusted-port** and **device-role switch** options). Network load is very high.	6 polling requests are being sent (2 hosts x 1 switch for *each* switch). Minimal network load. (Polling requests are sent by the local access switches to directly connected hosts, each polling request passes through fewer points in the network.)	6 polling requests are being sent (6 hosts x 1 switch) Network load is higher than secure zone 2, but not as high as secure zone 1. (Polling requests come from the distribution switch and go through the access switch before reaching the host.)

Secure Zone:	Secure Zone 1 - Inefficient and Unscalable Secure Zone	Secure Zone 2 - Efficient and Scalable Secure Zone When Binding Tables are Decentralized	Secure Zone 3: Efficient Secure Zone When Binding Table is Centralized
Efficiency:	Inefficient binding table, because the binding table is duplicated on each switch.	Efficient binding table, because each host's binding information is entered only once, and in one binding table and this the binding table of the directly connected switch.	Efficient binding table, because the binding information for each host is entered only once, and this is in the central binding table, which is on the distribution switch.
Recommended Action:	Reapply suitable policies to make the secure zone like secure zone 2	None; this is an efficient and scalable secure zone.	None; this is the best possible secure zone given the type of set-up (where the other switches in the network are either non-Cisco or do not support the feature)

Figure 105: Secure Zone 1 - Inefficient and Unscalable Secure Zone

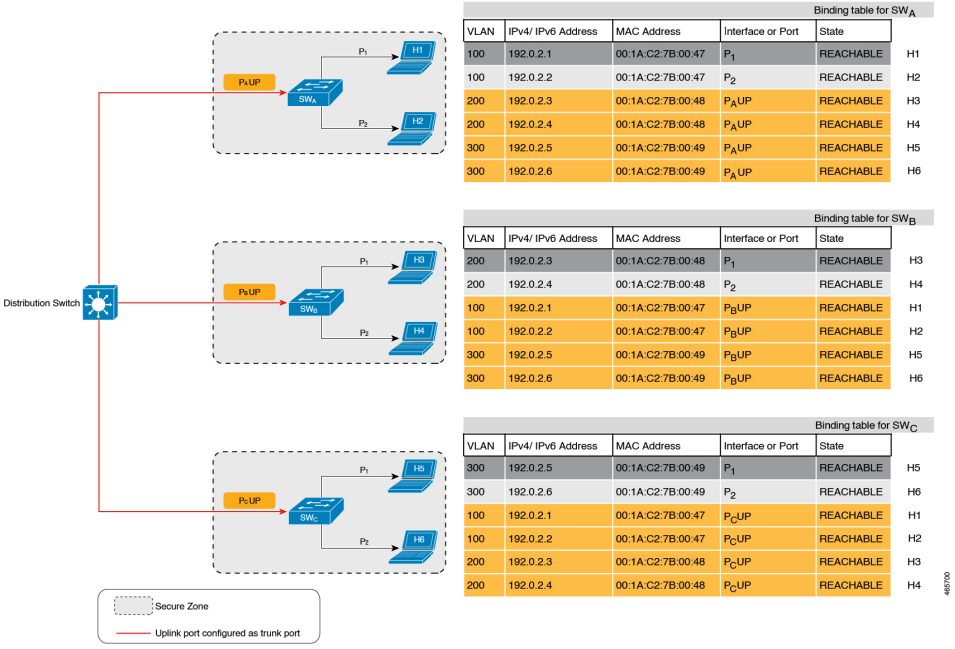
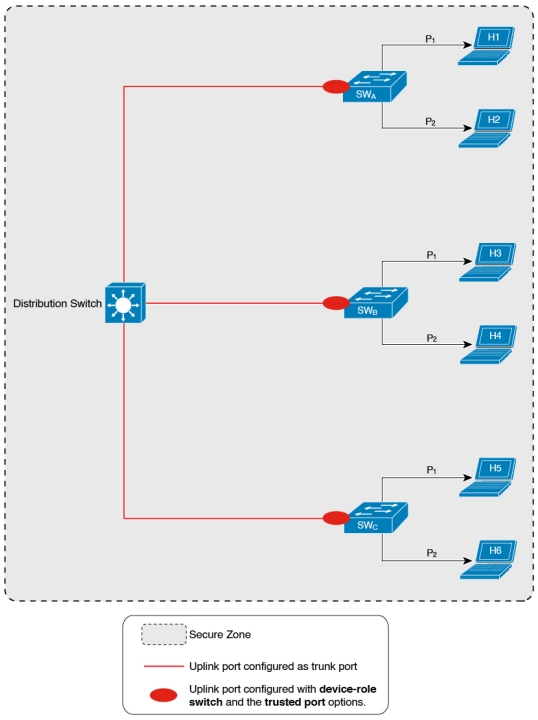


Figure 106: Secure Zone 2 - Efficient and Scalable Secure Zone When Binding Tables are Decentralized



Binding table for SW_A

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
100	192.0.2.1	00:1A:C2:7B:00:47	P_1	REACHABLE	H1
100	192.0.2.2	00:1A:C2:7B:00:47	P_2	**REACHABLE**	H2

Binding table for SW_B

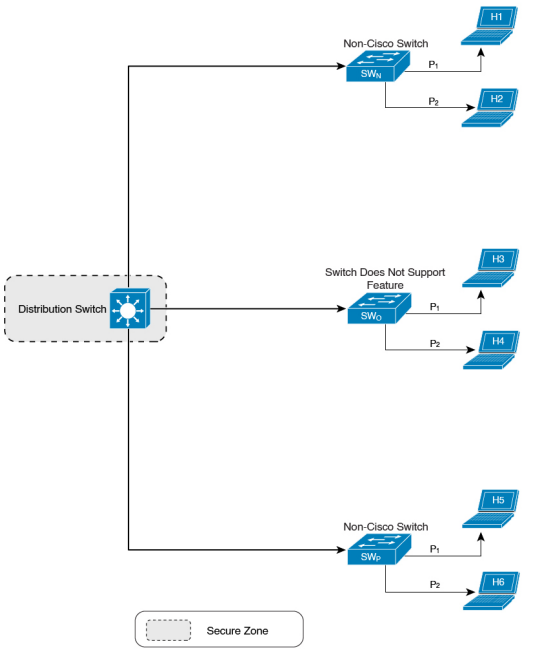
VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
200	192.0.2.3	00:1A:C2:7B:00:48	P_1	REACHABLE	H3
200	192.0.2.4	00:1A:C2:7B:00:48	P_2	REACHABLE	H4

Binding table for SW_C

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
300	192.0.2.5	00:1A:C2:7B:00:49	P_1	REACHABLE	H5
300	192.0.2.6	00:1A:C2:7B:00:49	P_2	REACHABLE	H6

465701

Figure 107: Secure Zone 3: Efficient Secure Zone When Binding Table is Centralized



Binding table for Distribution Switch

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
100	192.0.2.1	00:1A:C2:7B:00:47	P_NUP	REACHABLE	H1
100	192.0.2.2	00:1A:C2:7B:00:47	P_NUP	REACHABLE	H2
200	192.0.2.3	00:1A:C2:7B:00:48	P_OUP	REACHABLE	H3
200	192.0.2.4	00:1A:C2:7B:00:48	P_OUP	REACHABLE	H4
300	192.0.2.5	00:1A:C2:7B:00:49	P_PUP	REACHABLE	H5
300	192.0.2.6	00:1A:C2:7B:00:49	P_PUP	REACHABLE	H6

465702

When to Use Only Trusted-Port or Only Device-Role Switch

Configuring only **device-role switch** is suited to situations when you want to listen but not learn entries. For example, for Duplicate Address Detection (DAD), or when you want to send IPv6 or Neighbor Solicitation (NS) message on a switch-facing port.

When you configure this option on a switch port (or interface), SISF-based device-tracking treats the port as a trunk port, implying that the port is connected to other switches. It does not matter whether the port is actually a trunk port or not. Therefore, when NS packets or queries are sent to switches in the network for new entry validation, only the secure ports (ports where the **device-role switch** is configured) receive the packet or query. This safeguards the network. If the command is not configured on any port, a general broadcast of the query is sent.

Configuring only **trusted-port** is suited to situations where an access port should be configured as a trusted port. If an access port is connected to a DHCP server or a similar service that the switch is consuming, configuring an access port as a trusted port ensures that the service is not disrupted because traffic from such a port is trusted. This also widens the secure zone, to include the access port.

Address Count Limits

The address count limit parameter specifies limits for the number of IP and MAC addresses that can be entered in a binding table. The purpose of these limits is to contain the size of the binding table based on the number of known and expected hosts, thus enabling you to take pre-emptive action against rogue hosts or IPs in the network.

At a policy level there are separate limits for the number of IP addresses per port, the number of IPv4 addresses per MAC, and IPv6 addresses per MAC. You can configure or change only the number of IP addresses per port.

IP per Port

The IP per port option is the total number of IP addresses allowed for a port. The address can be IPv4 or IPv6. When the limit is reached, no further IP addresses (i.e., entries) are added to the binding table.

To configure this parameter in a policy, enter the **limit address-count** *ip-per-port* keyword in device-tracking configuration mode. If you configure a limit that is lower than the currently configured one, then the new (lower) limit is applicable only to new entries. An existing entry remains in the binding table and goes through its binding entry lifecycle.

IPv4 per MAC and IPv6 per MAC

This refers to the number of IPv4 addresses that can be mapped to one MAC address and the number of IPv6 addresses that can be mapped to one MAC address. When the limit is reached, no further entries can be added to the binding table, and traffic from new hosts will be dropped

Note The IPv4 per MAC limit and the IPv6 per MAC limit that is effective on an interface or VLAN is as defined in the policy that is applied. If the policy does not specify a limit, this means that a limit does not exist. You cannot change or configure a limit for IPv4 per MAC or IPv6 per MAC for any kind of policy (programmatic, or custom policy, or default policy).

Enter the **show device-tracking policy** *policy name* to check if a limit exists.

The following is sample output of a policy where an IPv4 per MAC and an IPv6 per MAC limit exists:

```
Device# show device-tracking policy LISP-DT-GUARD-VLAN
Policy LISP-DT-GUARD-VLAN configuration:
  security-level guard (*)
  <output truncated>

  limit address-count for IPv4 per mac 4 (*)
  limit address-count for IPv6 per mac 12 (*)
  tracking enable

<output truncated>
```

Address Count Limit Considerations and Interactions with Other SISF Settings

- The limits do not have a hierarchy, but the threshold that is set for each limit affects the others.

  For example, if the IP per port limit is 100, and the IPv4 per MAC limit is one, the limit is reached with a single host's IPv4-MAC binding entry. No further IP entries, which are bound to the same MAC are allowed in the table even though the port has a provision for 99 more IP addresses. Similarly, if the IP per port limit is one, and the IPv4 per MAC limit is 100. The limit is reached with a single host's IPv4-MAC binding entry. No further IP entries are allowed in the table even though the MAC has a provision for 99 more IP addresses for *that* MAC.

- Global and policy-level limits

  The limits configured with the **device-tracking binding max-entries** command are at the global level, the limits configured with the **limit address-count** command in the device-tracking configuration mode are for a policy, which is at the interface or VLAN level.

  If a policy-level value *and* a globally configured value exists, the creation of binding entries is stopped when *a* limit is reached - this limit can be any one of the global values or the policy-level value.

  If only globally configured values exist, the creation of binding entries is stopped when *a* limit is reached.

  If only a policy-level value exists, the creation of binding entries is stopped when the policy-level limit is reached.

Tracking

The tracking parameter involves tracking of hosts in the network. In section #unique_1993 unique_1993_Connect_42_section_axm_sbk_ctb above, this is referred to as "polling". It also describes polling behaviour in detail.

To configure polling parameters at the global level, enter the **device-tracking tracking** command in global configuration mode. After you configure this command you still have the flexibility to turn polling on or off, for individual interfaces and VLANs. For this you must enable or disable polling in the policy.

To enable polling in a policy, enter the **tracking enable** keywords in the device-tracking configuration mode. By default, polling is disabled in a policy.

Guidelines for Policy Creation

- If multiple policies are available on a given target, a system-internal policy priority determines which policy takes precedence.

  A manually created policy has the highest priority. When you want to override the settings of a programmatically created policy, you can create a custom policy, so it has higher priority.

- The parameters of a programmatically created policy cannot be changed. You can configure certain attributes of a custom policy.

Guidelines for Applying a Policy

- Multiple policies can be attached to the same VLAN.
- If a programmatic policy is attached to a VLAN and you want to change policy settings, create a custom device-tracking policy and attach it to the VLAN.
- When multiple policies with different priorities are attached to the same VLAN, the settings of the policy with the highest priority are effective. The exceptions here are the limit address-count for IPv4 per mac and limit address-count for IPv6 per mac settings - the settings of the policy with the lowest priority are effective.
- When a device-tracking policy is attached to an interface under a VLAN, the policy settings on the interface take precedence over those on its VLAN; exceptions here are the values for limit address-count for IPv4 per mac and limit address-count for IPv6 per mac, which are aggregated from the policy on both the interface and VLAN.
- A policy cannot be removed unless the device tracking client feature configuration is removed.

How to Configure SISF

SISF or SISF-based device-tracking, is disabled by default. You enable it by defining a device-tracking policy and attaching the policy to a specific target. The target could be an interface or a VLAN. There are multiple ways to define a policy and no single method is a preferred or recommended one - use the option that suits your requirements.

Method of Enabling SISF	Applicable Configuration Tasks	Result
Option 1: Manually, by using interface configuration commands to create and apply the **default** policy to a target.	Applying the Default Device Tracking Policy to a Target, on page 1405	Automatically applies the **default** device tracking policy to the specified target. The **default** policy is a built-in policy with default settings; you cannot change any of the attributes of the default policy. See **Option 2** if you want to configure device tracking policy attributes.

Method of Enabling SISF	Applicable Configuration Tasks	Result
Option 2: Manually, by using global configuration commands to create a custom policy and applying the custom policy to a target.	1. Creating a Custom Device Tracking Policy with Custom Settings, on page 1405 2. Attach the custom policy to an interface or VLAN: Attaching a Device Tracking Policy to an Interface, on page 1409 OR Attaching a Device Tracking Policy to a VLAN, on page 1410	Creates a custom policy with the name and policy parameters you configure, and attaches the policy to the specified target.
Option 3: Programmatically, by configuring the snooping command.	Enter the **ip dhcp snooping vlan** *vlan* command in global configuration mode. Example: Programatically Enabling SISF by Configuring DHCP Snooping, on page 1413	When you configure the command, the system automatically creates policy `DT-PROGRAMMATIC`. Use this method if you want to enable SISF-based device tracking for these clients: IEEE 802.1X, Web authentication, Cisco TrustSec, IP Source Guard, and SANET.
Option 4: Programmatically, by configuring Locator ID Separation Protocol (LISP).	Example: Programatically enabling SISF by Configuring LISP (LISP-DT-GUARD-VLAN), on page 1415 Example: Programatically Enabling SISF by Configuring LISP (LISP-DT-GLEAN-VLAN), on page 1414	When you configure LISP, the system automatically creates policy `LISP-DT-GUARD-VLAN` or `LISP-DT-GLEAN-VLAN`.
Option 5: Programmatically, by configuring EVPN VLAN.	Example: Programatically Enabling SISF by Configuring EVPN on VLAN, on page 1414	When you configure EVPN on VLAN, the system automatically creates policy `evpn-sisf-policy`.
Option 6: By using an interface template	Using an Interface Template to Enable SISF, on page 1411	By adding the policy to an interface template, you can apply the same policy to multiple targets, without having to create it separately for each target.
Option 7: Migrating from legacy IPDT and IPv6 Snooping.	Migrating from Legacy IPDT and IPv6 Snooping to SISF-Based Device-Tracking, on page 1412	Convert legacy IPDT and IPv6 Snooping configuration to the SISF-based device-tracking commands.

Applying the Default Device Tracking Policy to a Target

Beginning in privileged EXEC mode, follow these steps to apply the default device tracking policy to an interface or VLAN:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	Specify an interface or a VLAN • **interface** *type number* • **vlan configuration** *vlan_list* **Example:** `Device(config)# interface gigabitethernet 1/1` OR `Device(config)# vlan configuration 333`	**interface** *type number*—Specifies the interface and enters interface configuration mode. The device tracking policy will be attached to the specified interface. **vlan configuration** *vlan_list*—Specifies the VLANs and enters VLAN feature configuration mode. The device tracking policy will be attached to the specified VLAN.
Step 4	**device-tracking** **Example:** `Device(config-if)# device-tracking` OR `Device(config-vlan-config)# device-tracking`	Enables SISF-based device tracking and attaches the default policy it to the interface or VLAN. The default policy is a built-in policy with default settings; none of the attributes of the default policy can be changed.
Step 5	**end** **Example:** `Device(config-if)# end` OR `Device(config-vlan-config)# end`	Exits interface configuration mode and returns to privileged EXEC mode. Exits VLAN feature configuration mode and returns to privileged EXEC mode.
Step 6	**show device-tracking policy** *policy-name* **Example:** `Device# show device-tracking policy default`	Displays device-tracking policy configuration, and all the targets it is applied to.

Creating a Custom Device Tracking Policy with Custom Settings

Beginning in privileged EXEC mode, follow these steps to create and configure a device tracking policy:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	[**no**] **device-tracking policy** *policy-name* **Example:** `Device(config)# device-tracking policy example_policy`	Creates the policy and enters device-tracking configuration mode.
Step 3	[**data-glean** | **default** | **destination-glean** | **device-role** | **distribution-switch** | **exit** | **limit** | **no** | **prefix-glean** | **protocol** | **security-level** | **tracking** | **trusted-port** | **vpc**] **Example:** `Device(config-device-tracking)# destination-glean log-only`	Enter the question mark (?) at the system prompt to obtain a list of available options in this mode. You can configure the following for both IPv4 and IPv6: • (Optional) **data-glean**—Enables learning of addresses from a data packet snooped from a source inside the network and populates the binding table with the data traffic source address. Enter one of these options: • log-only—Generates a syslog message upon data packet notification • recovery—Uses a protocol to enable binding table recovery. Enter **NDP** or **DHCP**. • (Optional) **default**—Sets the policy attribute to its default value. You can set these policy attributes to their default values: **data-glean**, **destination-glean**, **device-role**, **limit**, **prefix-glean**, **protocol**, **security-level**, **tracking**, **trusted-port**. • (Optional) **destination-glean**—Populates the binding table by gleaning data traffic destination address. Enter one of these options: • **log-only**—Generates a syslog message upon data packet notification • **recovery**—Uses a protocol to enable binding table recovery. Enter **DHCP**. • (Optional) **device-role**—Sets the role of the device attached to the port. It can be a

	Command or Action	Purpose
		node or a switch. Enter one of these options: • node—Configures the attached device as a node. This is the default option. • switch—Configures the attached device as a switch. • (Optional) **distribution-switch**—Although visible on the CLI, this option is not supported. Any configuration settings you make will not take effect. • **exit**—Exits the device-tracking policy configuration mode. • **limit** address-count—Specifies an address count limit per port. The range is 1 to 32000. • **no**—Negates the command or sets it to defaults. • (Optional) **prefix-glean**—Enables learning of prefixes from either IPv6 Router Advertisements or from DHCP-PD. You have the following option: • (Optional) **only**—Gleans only prefixes and not host addresses. • (Optional) **protocol**—Sets the protocol to glean; by default, all are gleaned. Enter one of these options: • **arp** [**prefix-list** *name*]: Gleans addresses in ARP packets. Optionally, enter the name of prefix-list that is to be matched. • **dhcp4** [**prefix-list** *name*]: Glean addresses in DHCPv4 packets. Optionally, enter the name of prefix-list that is to be matched. • **dhcp6** [**prefix-list** *name*]: Glean addresses in DHCPv6 packets. Optionally, enter the name of prefix-list that is to be matched. • **ndp** [**prefix-list** *name*]: Glean addresses in NDP packets.

	Command or Action	Purpose
		Optionally, enter the name of prefix-list that is to be matched. • (Optional) **security-level**—Specifies the level of security enforced by the feature. Enter one of these options: • **glean**—Gleans addresses passively. • **guard**—Inspects and drops un-authorized messages. This is the default. • **inspect**—Gleans and validates messages. • (Optional) **tracking**—Specifies a tracking option. Enter one of these options: • **disable** [**stale-lifetime** [*1-86400-seconds* | **infinite**]]—Turns of device-tracking. Optionally, you can enter the duration for which the entry is kept inactive before deletion, or keep it permanently inactive. • **enable** [**reachable-lifetime** [*1-86400-seconds* | **infinite**]]—Turns on device-tracking. Optionally, you can enter the duration for which the entry is kept reachable, or keep it permanently reachable. • (Optional) **trusted-port**—Sets up a trusted port. Disables the guard on applicable targets. Bindings learned through a trusted port have preference over bindings learned through any other port. A trusted port is given preference in case of a collision while making an entry in the table. • (Optional) **vpc**—Although visible on the CLI, this option is not supported. Any configuration settings you make will not take effect.
Step 4	**end** **Example:** `Device(config-device-tracking)# end`	Exits device-tracking configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 5	**show device-tracking policy** *policy-name* **Example:** `Device# show device-tracking policy example_policy`	Displays the device-tracking policy configuration.

What to do next

Attach the policy to an interface or VLAN.

Attaching a Device Tracking Policy to an Interface

Beginning in privileged EXEC mode, follow these steps to attach a device tracking policy to an interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface* **Example:** `Device(config-if)# interface gigabitethernet 1/1`	Specifies an interface and enters interface configuration mode.
Step 4	**device-tracking attach-policy** *policy name* **Example:** `Device(config-if)# device-tracking attach-policy example_policy`	Attaches the device tracking policy to the interface. Device tracking is also supported on EtherChannels. **Note** SISF based device-tracking policies can be disabled only if they are custom policies. Programmatically created policies can be removed only if the corresponding device-tracking client feature configuration is removed.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show device-tracking policies** [**interface** *interface*] **Example:** `Device# show device-tracking policies interface gigabitethernet 1/1`	Displays policies that match the specified interface type and number.

Attaching a Device Tracking Policy to a VLAN

Beginning in privileged EXEC mode, follow these steps to attach a device-tracking policy to VLANs across multiple interfaces:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan configuration** *vlan_list* **Example:** `Device(config)# vlan configuration 333`	Specifies the VLANs to which the device tracking policy will be attached; enters the VLAN interface configuration mode.
Step 4	**device-tracking attach-policy** *policy_name* **Example:** `Device(config-vlan-config)# device-tracking attach-policy example_policy`	Attaches the device tracking policy to the specified VLANs across all switch interfaces. **Note** SISF based device-tracking policies can be disabled only if they are custom policies. Programmatically created policies can be removed only if the corresponding device-tracking client feature configuration is removed.
Step 5	**do show device-tracking policies vlan** *vlan-ID* **Example:** `Device(config-vlan-config)# do show device-tracking policies vlan 333`	Verifies that the policy is attached to the specified VLAN, without exiting the VLAN interface configuration mode.
Step 6	**end** **Example:**	Exits VLAN feature configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-vlan-config)# end`	

Using an Interface Template to Enable SISF

An interface template is a container of configurations or policies. When you apply the interface template to a target, all the configurations are applied to the target. This enables you to configure multiple commands or features on one or more targets.

You can add the **device-tracking policy** *policy_name* global configuration command to an interface template. SISF-based device-tracking is enabled and the policy is applied, wherever the template is applied.

You can also apply the template through 802.1x authentication. During the 802.1x authentication process, you can dynamically assign different templates (and therefore different policies) to different interfaces.

Note You can apply only one interface template to one port.

Before you begin

You have already created a custom policy. See Creating a Custom Device Tracking Policy with Custom Settings, on page 1405.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**template interface** *template_name* **Example:** `Device(config)# template interface template_w_sisf`	Creates a template with the name you specify and enters template configuration mode. In the accompanying example, a template called "template_w_sisf" is created.
Step 4	**device-tracking attach-policy** *policy_name* **Example:** `Device (config-template)# device-tracking attach-policy sisf_policy_for_template`	Attaches a policy to the template. SISF-based device-tracking is enabled and the policy is applied wherever the template is applied.
Step 5	**exit** **Example:**	Exits the template configuration mode and enters the global configuration mode.

	Command or Action	Purpose
	`Device (config-template)# exit`	
Step 6	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies an interface and enters interface configuration mode.
Step 7	**source template** *template_name* **Example:** `Device(config-if)# source template template_w_sisf`	Configures a static binding for an interface template. In the accompanying example, "template_w_sisf" is statically applied to an interface.
Step 8	**end** **Example:** `Device(config-if)# end`	Exits the interface configuration mode and enters the privileged EXEC mode.
Step 9	**show running-config interface** *type number* **Example:** `Device# show running-config interface gigabitethernet 1/1` `Building configuration...` `<output truncated>` `Current configuration : 71 bytes` `!` `interface GigabitEthernet1/1` `source template template_w_sisf` `end` `<output truncated>`	Displays the contents of the running configuration.

Migrating from Legacy IPDT and IPv6 Snooping to SISF-Based Device-Tracking

Based on the legacy configuration that exists on your device, the **device-tracking upgrade-cli** global configuration command upgrades your CLI differently. Consider the following configuration scenarios and the corresponding migration results before you migrate your existing configuration.

Note You cannot configure a mix of the old IPDT and IPv6 Snooping commands with the SISF-based device-tracking commands.

Only IPDT Configuration Exists

If your device has only IPDT configuration, running the **device-tracking upgrade-cli** command converts the configuration to use a SISF policy that is created and attached to the interface. You can then update this SISF policy.

If you continue to use the legacy commands, this restricts you to operate in a legacy mode where only the legacy IPDT and IPv6 Snooping commands are available on the device.

Only IPv6 Snooping Configuration Exists

On a device with existing IPv6 Snooping configuration, the old IPv6 Snooping commands are available for further configuration. The following options are available:

- (Recommended) Use the **device-tracking upgrade-cli** command to convert all your legacy configuration to the SISF-based device-tracking commands. After conversion, only the SISF-based device-tracking commands will work on your device.
- Use the legacy IPv6 Snooping commands for your future configuration and do not run the **device-tracking upgrade-cli** command. With this option, only the legacy IPv6 Snooping commands are available on your device, and you cannot use the SISF-based device-tracking commands.

Both IPDT and IPv6 Snooping Configuration Exist

On a device that has both legacy IPDT configuration and IPv6 Snooping configuration, you can convert legacy commands to the SISF-based device-tracking commands. However, note that only one snooping policy can be attached to an interface, and the IPv6 Snooping policy parameters override the IPDT settings.

Note If you do not migrate to the SISF-based device-tracking commands and continue to use the legacy IPv6 Snooping or IPDT commands, your IPv4 device-tracking configuration information may be displayed in the IPv6 Snooping commands, as the SISF-based device-tracking feature handles both IPv4 and IPv6 configuration. To avoid this, we recommend that you convert your legacy configuration to SISF-based device-tracking commands.

No IPDT or IPv6 Snooping Configuration Exists

If your device has no legacy IP Device Tracking or IPv6 Snooping configurations, you can use only the SISF-based device-tracking commands for all your future configuration. The legacy IPDT commands and IPv6 Snooping commands are not available.

Configuration Examples for SISF

Example: Programatically Enabling SISF by Configuring DHCP Snooping

The following example shows how to configure the **ip dhcp snooping vlan** *vlan* command in global configuration mode to enable SISF-based device-tracking. When you enable SISF this way, the system creates the `DT-PROGRMMATIC` policy.

Enter the **show device-tracking policy** *policy_name* command in privileged EXEC mode, to display the settings for a `DT-PROGRMMATIC` policy.

```
Device> enable
Device# configure terminal
Device(config)# ip dhcp snooping vlan 10
Device(config)# end
```

```
Device# show device-tracking policy DT-PROGRAMMATIC

Policy DT-PROGRAMMATIC configuration:
  security-level glean (*)
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  limit address-count for IPv4 per mac 1 (*)
  tracking enable
Policy DT-PROGRAMMATIC is applied on the following targets:
Target      Type    Policy               Feature           Target range
vlan 10     VLAN    DT-PROGRAMMATIC      Device-tracking   vlan all


  note:
  Binding entry Down timer: 24 hours (*)
  Binding entry Stale timer:   24 hours (*)
```

Example: Programatically Enabling SISF by Configuring EVPN on VLAN

When you configure EVPN, the system automatically creates programmatic policy `evpn-sisf-policy`. Enter the **show device-tracking policy** *policy_name* command in privileged EXEC mode, to display policy settings.

```
Device# show device-tracking policy evpn-sisf-policy

Policy evpn-sisf-policy configuration:
  security-level glean (*)
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  tracking enable
Policy evpn-sisf-policy is applied on the following targets:
Target      Type    Policy                  Feature           Target range
vlan 10     VLAN    evpn-sisf-policy        Device-tracking   vlan all
  note:
  Binding entry Down timer: 24 hours (*)
  Binding entry Stale timer:   24 hours (*)
```

Example: Programatically Enabling SISF by Configuring LISP (LISP-DT-GLEAN-VLAN)

The following is sample output of programmatic policy `LISP-DT-GLEAN-VLAN`.

Note The system creates `LISP-DT-GUARD-VLAN`, or `LISP-DT-GLEAN-VLAN`, or `LISP-DT-GUARD-VLAN-MULTI-IP` depending on *how* LISP is configured. You cannot change this, but if required you can create a custom policy with custom settings and attach it to the required target.

To display policy settings, enter the **show device-tracking policy** *policy_name* command in privileged EXEC mode.

```
Device# show device-tracking policy LISP-DT-GLEAN-VLAN

Policy LISP-DT-GLEAN-VLAN configuration:
  security-level glean (*)
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  limit address-count for IPv4 per mac 4 (*)
  limit address-count for IPv6 per mac 12 (*)
  tracking enable
Policy LISP-DT-GUARD-VLAN is applied on the following targets:
Target     Type   Policy                 Feature            Target range
vlan 10    VLAN   LISP-DT-GLEAN-VLAN     Device-tracking    vlan all

  note:
  Binding entry Down timer: 10 minutes (*)
  Binding entry Stale timer:   30 minutes (*)
```

Example: Programatically enabling SISF by Configuring LISP (LISP-DT-GUARD-VLAN)

The following is sample output of programmatic policy `LISP-DT-GUARD-VLAN`.

Note The system creates `LISP-DT-GUARD-VLAN`, or `LISP-DT-GLEAN-VLAN`, or `LISP-DT-GUARD-VLAN-MULTI-IP` depending on *how* LISP is configured. You cannot change this, but if required you can create a custom policy with custom settings and attach it to the required target.

To display policy settings, enter the **show device-tracking policy** *policy_name* command in privileged EXEC mode.

```
Device# show device-tracking policy LISP-DT-GUARD-VLAN

Policy LISP-DT-GUARD-VLAN configuration:
  security-level guard (*)
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  limit address-count for IPv4 per mac 4 (*)
  limit address-count for IPv6 per mac 12 (*)
  tracking enable
Policy LISP-DT-GUARD-VLAN is applied on the following targets:
Target      Type    Policy                   Feature            Target range
vlan 10     VLAN    LISP-DT-GUARD-VLAN       Device-tracking    vlan all
  note:
  Binding entry Down timer: 10 minutes (*)
  Binding entry Stale timer:   30 minutes (*)
```

Example: Mitigating the IPv4 Duplicate Address Problem

This example shows how you can tackle the `Duplicate IP Address 0.0.0.0 error message` problem encountered by clients that run Microsoft Windows:

Configure the **device-tracking tracking auto-source** command in global configuration mode. This command determines the source IP and MAC address used in the ARP probe sent by the switch to probe a client, in order to maintain its entry in the device-tracking table. The purpose, is to avoid using 0.0.0.0 as source IP address.

Note Configure the **device-tracking tracking auto-source** command when a switch virtual interface (SVI) is not configured. You do not have to configure it when a SVI is configured with an IPv4 address on the VLAN.

Command	Action **(In order to select source IP and MAC address for device tracking ARP probe)**	Notes
device-tracking tracking auto-source global configuration command.	• Set source to VLAN SVI if present. • Look for IP and MAC binding in device-tracking table from same subnet. • Use 0.0.0.0	We recommend that you disable device-tracking on all trunk ports to avoid MAC flapping.
device-tracking tracking auto-source override global configuration command.	• Set source to VLAN SVI if present • Use 0.0.0.0	Not recommended when there is no SVI.
device-tracking tracking auto-source fallback 0.0.0.X 255.255.255.0 global configuration command.	• Set source to VLAN SVI if present. • Look for IP and MAC binding in device-tracking table from same subnet. • Compute source IP from client IP using host bit and mask provided. Source MAC is taken from the MAC address of the switchport facing the client*.	We recommend that you disable device-tracking on all trunk ports to avoid MAC flapping. The computed IPv4 address must not be assigned to any client or network device.

Command	Action (In order to select source IP and MAC address for device tracking ARP probe)	Notes
device-tracking tracking auto-source fallback 0.0.0.X 255.255.255.0 override global configuration command.	• Set source to VLAN SVI if present. Compute source IP from client IP using host bit and mask provided*. Source MAC is taken from the MAC address of the switchport facing the client*.	-

* Depending on the client IP address, an IPv4 address has to be reserved for the source IP.

A reserved source IPv4 address = (host-ip and mask) | client-ip

- Client IP = 192.0.2.25
- Source IP = (192.0.2.25 and 255.255.255.0) | (0.0.0.1) = 192.0.2.1

IP address 192.0.2.1 should not be assigned to any client or network device.

Example: Disabling IPv6 Device Tracking on a Target

By default, SISF-based device-tracking supports both IPv4 and IPv6. The following configuration examples show how you can disable IPv6 device-tracking where supported.

To disable device-tracking for IPv6, when a *custom* policy is attached to a target (all releases):

```
Device(config)# device-tracking policy example-policy
 Device(config-device-tracking)# no protocol ndp
 Device(config-device-tracking)# no protocol dhcp6
 Device(config-device-tracking)# end
```

Note You cannot change the settings of a programmatic policy

Example: Enabling IPv6 for SVI on VLAN (To Mitigate the Duplicate Address Problem)

When IPv6 is enabled in the network and a switched virtual interface (SVI) is configured on a VLAN, we recommend that you add the following to the SVI configuration. This enables the SVI to acquire a link-local address automatically; this address is used as the source IP address of the SISF probe, thus preventing the duplicate IP address issue.

```
Device> enable
Device# configure terminal
Device(config)# interface vlan 10
```

```
Device(config-if)# ipv6 enable
Device(config-if)# end
```

Example: Configuring a Multi-Switch Network to Stop Creating Binding Entries from a Trunk Port

In a multi-switch network, SISF-based device tracking provides the capability to distribute binding table entries between switches running the feature. Binding entries are only created on the switches where the host appears on an access port. No entry is created for a host that appears over a trunk port. This is achieved by configuring a policy with the **trusted-port** and **device-role switch** options, and attaching it to the trunk port.

Note Both, the **trusted-port**, and **device-role switch** options, must be configured in the policy.

Further, we recommended that you apply such a policy on a port facing a device, which also has SISF-based device tracking enabled.

```
Device> enable
Device# configure terminal
Device(config)# device-tracking policy example_trusted_policy
Device(config-device-tracking)# device-role switch
Device(config-device-tracking)# trusted-port
Device(config-device-tracking)# exit
Device(config)# interface gigabitethernet 1/1
Device(config-if)# device-tracking attach-policy  example_trusted_policy
Device(config-if)# end
```

Example: Avoiding a Short Device-Tracking Binding Reachable Time

When migrating from an older release, the following configuration may be present:

```
device-tracking binding reachable-lifetime 10
```

Remove this by entering the **no** version of the command.

```
Device> enable
Device# configure terminal
Device(config)# no device-tracking binding reachable-lifetime 10
Device(config)# end
```

Example: Detecting and Preventing Spoofing

Address spoofing, is a man-in-the-middle attack that allows an attacker to intercept communication between network devices. These attacks attempt to divert traffic from its originally intended host to the attacker instead. For example, attacks are carried out by sending unsolicited Address Resolution Protocol (ARP) replies or with IPv6 Neighbor Advertisements carrying a mapping that is different from the legitimate one, such as <IPTARGET, MACTHIEF>. When the IPTARGET is of the default gateway, all traffic that is meant to leave the subnet is routed to the attacker.

The following example shows the required SISF configuration to enable the system to detect and prevent spoofing. It also shows the system messages that are logged when spoofing is detected, and the action that

the system takes. It includes an excerpt of LISP configuration in an SDA setup for example purposes only. Actual LISP configuration may involve additional configuration.

Sample LISP configuration:

```
instance-id 100
  service ethernet
   eid-table vlan 100               <<< triggers creation of programmatic policy
“LISP-DT-GUARD-VLAN”
   database-mapping mac locator-set XTR11
   exit-service-ethernet
  !
  exit-instance-id
```

Settings of the programmatic policy:

```
Device# show device-tracking policy LISP-DT-GUARD-VLAN
Device-tracking policy LISP-DT-GUARD-VLAN configuration:
  security-level guard              <<< enables the detection and prevention of IPv4 and
IPv6 spoofing
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  limit address-count for IPv4 per mac 21
  limit address-count for IPv6 per mac 58
  origin fabric
  tracking enable reachable-lifetime 240
```

The following device-tracking counters show you that packet drops have occurred. However, the drops may be caused by reasons other than address spoofing as well. Use the information in the counters along with system messages to ascertain if spoofing has occurred.

```
Device# show device-tracking counters vlan 11
Received messages on vlan 11   :
Protocol        Protocol message
NDP             RS[4] RA[4] NS[1777] NA[2685]
DHCPv6
ARP             REQ[12] REP[1012]
DHCPv4
ACD&DAD         --[8]
:
Dropped messages on vlan 10   :
Feature             Protocol Msg [Total dropped]
Device-tracking:    ARP      REQ [23]
                    reason:  Packet accepted but not forwarded [23]
                             REP [450]
                    reason:  Silent drop [445]
                    reason:  Packet accepted but not forwarded [5] :
```

Required configuration to display system messages:

```
Device# device-tracking logging theft
Device# device-tracking logging packet drop
```

While the packet drops in the device-tracking counters do not conclusively prove that spoofing has occurred, the system messages help you ascertain this.

```
%SISF-4-IP_THEFT: IP Theft IP=3001::5 VLAN=10 Cand-MAC=aabb.cc00.6600 Cand-I/F=Et0/0 Known
 MAC aabb.cc00.6900 Known I/F Et0/1
%SISF-4-IP_THEFT: IP Theft IP=FE80::A8BB:CCFF:FE00:6900 VLAN=10 Cand-MAC=aabb.cc00.6600
Cand-I/F=Et0/0 Known MAC aabb.cc00.6900 Known I/F Et0/1
```

In the log, verified binding information (IP, MAC address, interface or VLAN) is preceded by the term "Known" . A suspicious IP address and MAC address is preceded by the term "New" or "Cand". Interface and VLAN information is also provided along with the suspicious IP or MAC address - this helps you identify where the suspicious traffic was seen.

For more information about how to interpret these system messages, in the command reference of the corresponding release, see the usage guidelines of the **device-tracking logging** command.

CHAPTER 100

Configuring IEEE 802.1x Port-Based Authentication

This chapter describes how to configure IEEE 802.1x port-based authentication. IEEE 802.1x authentication prevents unauthorized devices (clients) from gaining access to the network. Unless otherwise noted, the term *switch* refers to a standalone switch.

Information About IEEE 802.1x Port-Based Authentication

The 802.1x standard defines a client-server-based access control and authentication protocol that prevents unauthorized clients from connecting to a LAN through publicly accessible ports unless they are properly authenticated. The authentication server authenticates each client connected to a switch port before making available any services offered by the switch or the LAN.

Note TACACS is not supported with 802.1x authentication.

Until the client is authenticated, 802.1x access control allows only Extensible Authentication Protocol over LAN (EAPOL), Cisco Discovery Protocol, and Spanning Tree Protocol (STP) traffic through the port to which the client is connected. After authentication is successful, normal traffic can pass through the port.

The table shown below lists the maximum number of session each client supports:

Client session	Maximum sessions supported
Maximum dot1x or MAB client sessions	2000
Maximum web-based authentication sessions	2000
Maximum dot1x sessions with critical-auth VLAN enabled and server re-initialized	2000
Maximum MAB sessions with various session features applied	2000

Client session	Maximum sessions supported
Maximum dot1x sessions with service templates or session features applied	2000

Overview of IEEE 802.1x Port-Based Authentication

The 802.1x standard defines a client-server-based access control and authentication protocol that prevents unauthorized clients from connecting to a LAN through publicly accessible ports unless they are properly authenticated. The authentication server authenticates each client connected to a switch port before making available any services offered by the switch or the LAN.

Note TACACS is not supported with 802.1x authentication.

Until the client is authenticated, 802.1x access control allows only Extensible Authentication Protocol over LAN (EAPOL), Cisco Discovery Protocol, and Spanning Tree Protocol (STP) traffic through the port to which the client is connected. After authentication is successful, normal traffic can pass through the port.

Port-Based Authentication Process

To configure IEEE 802.1X port-based authentication, you must enable authentication, authorization, and accounting (AAA) and specify the authentication method list. A method list describes the sequence and authentication method to be queried to authenticate a user.

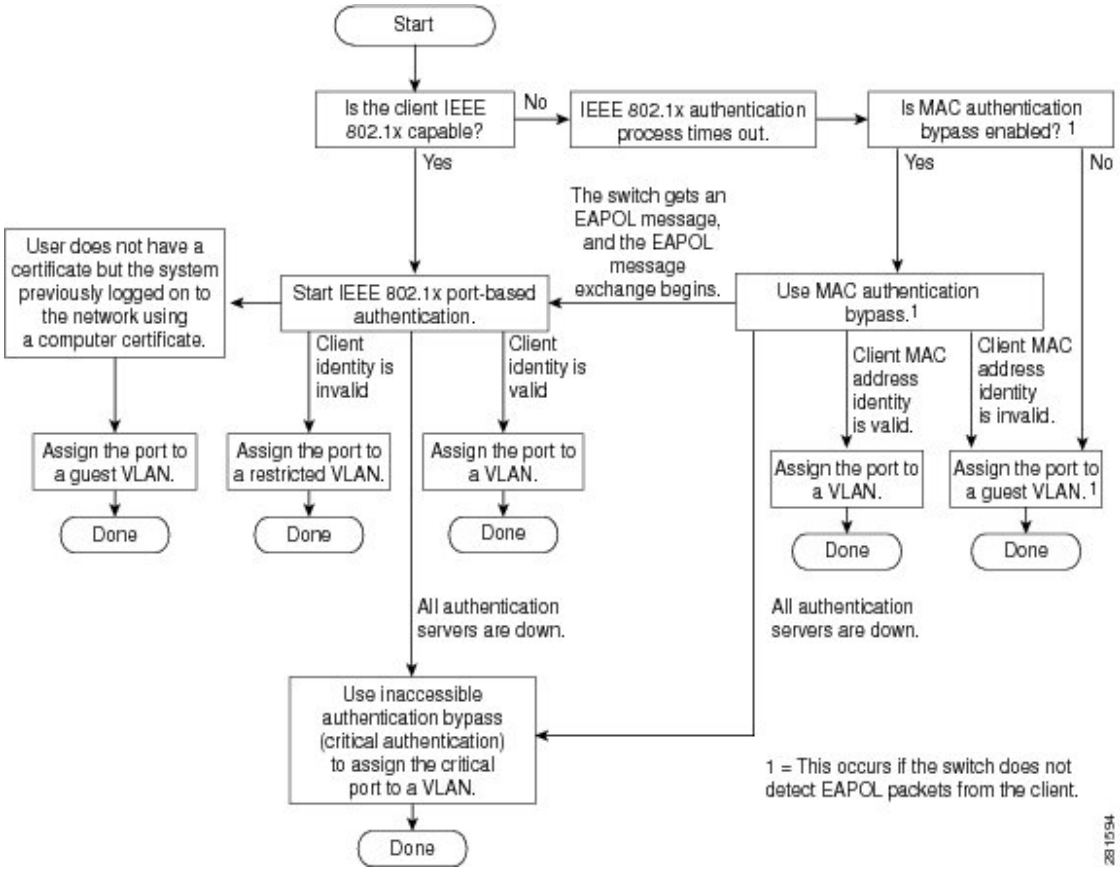
The AAA process begins with authentication. When 802.1x port-based authentication is enabled and the client supports 802.1x-compliant client software, these events occur:

- If the client identity is valid and the 802.1x authentication succeeds, the switch grants the client access to the network.
- If 802.1x authentication times out while waiting for an EAPOL message exchange and MAC authentication bypass is enabled, the switch can use the client MAC address for authorization. If the client MAC address is valid and the authorization succeeds, the switch grants the client access to the network. If the client MAC address is invalid and the authorization fails, the switch assigns the client to a guest VLAN that provides limited services if a guest VLAN is configured.
- If the switch gets an invalid identity from an 802.1x-capable client and a restricted VLAN is specified, the switch can assign the client to a restricted VLAN that provides limited services.
- If the RADIUS authentication server is unavailable (down) and inaccessible authentication bypass is enabled, the switch grants the client access to the network by putting the port in the critical-authentication state in the RADIUS-configured or the user-specified access VLAN.

Note Inaccessible authentication bypass is also referred to as critical authentication or the AAA fail policy.

If Multi Domain Authentication (MDA) is enabled on a port, this flow can be used with some exceptions that are applicable to voice authorization.

Figure 108: Authentication Flowchart



The switch reauthenticates a client when one of these situations occurs:

- Periodic reauthentication is enabled, and the reauthentication timer expires.

  You can configure the reauthentication timer to use a switch-specific value or to be based on values from the RADIUS server.

  After 802.1x authentication using a RADIUS server is configured, the switch uses timers based on the Session-Timeout RADIUS attribute (Attribute[27]) and the Termination-Action RADIUS attribute (Attribute [29]).

  The Session-Timeout RADIUS attribute (Attribute[27]) specifies the time after which reauthentication occurs. The range is 1 to 1073741823 seconds.

Note The supported timeout range is 1 to 65535 seconds.

The Termination-Action RADIUS attribute (Attribute [29]) specifies the action to take during reauthentication. The actions are *Initialize* and *ReAuthenticate*. When the *Initialize* action is set (the attribute value is *DEFAULT*), the 802.1x session ends, and connectivity is lost during reauthentication. When the *ReAuthenticate* action is set (the attribute value is RADIUS-Request), the session is not affected during reauthentication.

- You manually reauthenticate the client by entering the **dot1x re-authenticate interface** *interface-id* privileged EXEC command.

Port-Based Authentication Initiation and Message Exchange

During 802.1x authentication, the switch or the client can initiate authentication. If you enable authentication on a port by using the **authentication port-control auto** interface configuration command, the switch initiates authentication when the link state changes from down to up or periodically as long as the port remains up and unauthenticated. The switch sends an EAP-request/identity frame to the client to request its identity. Upon receipt of the frame, the client responds with an EAP-response/identity frame.

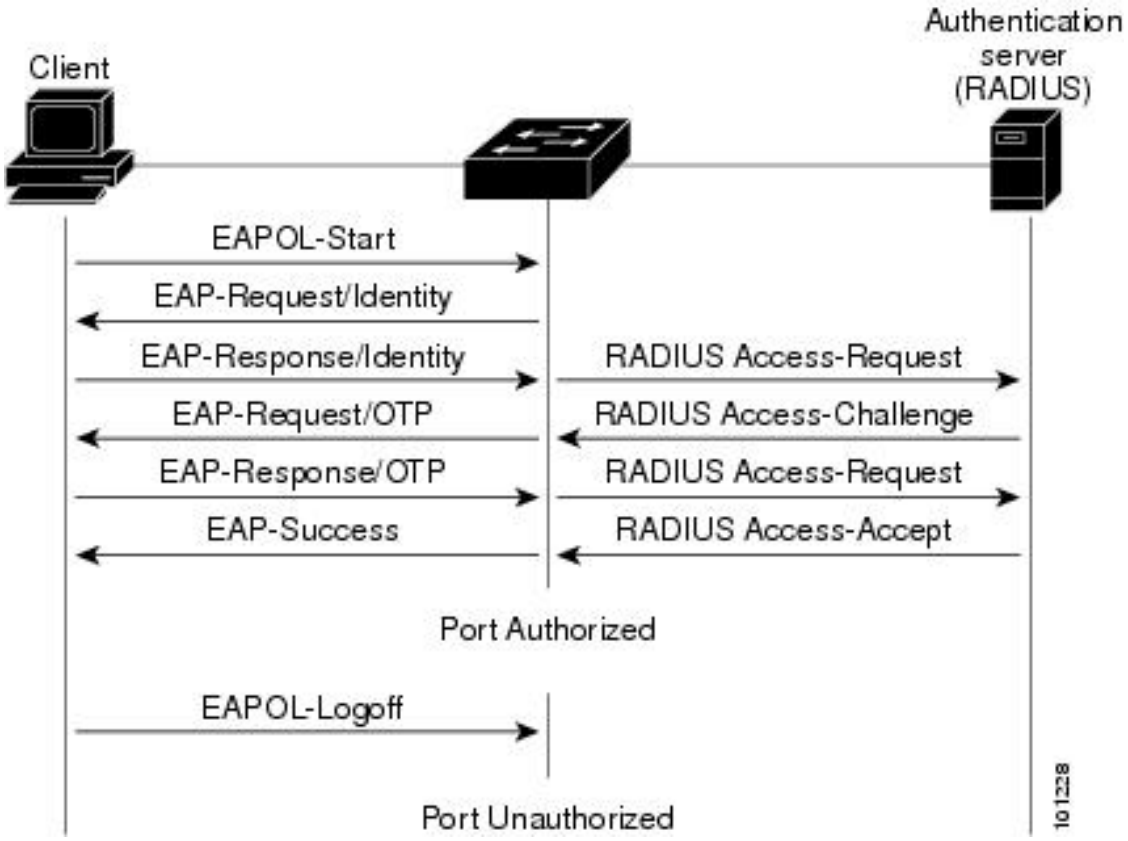
However, if during bootup, the client does not receive an EAP-request/identity frame from the switch, the client can initiate authentication by sending an EAPOL-start frame, which prompts the switch to request the client's identity.

Note If 802.1x authentication is not enabled or supported on the network access device, any EAPOL frames from the client are dropped. If the client does not receive an EAP-request/identity frame after three attempts to start authentication, the client sends frames as if the port is in the authorized state. A port in the authorized state effectively means that the client has been successfully authenticated.

When the client supplies its identity, the switch begins its role as the intermediary, passing EAP frames between the client and the authentication server until authentication succeeds or fails. If the authentication succeeds, the switch port becomes authorized. If the authentication fails, authentication can be retried, the port might be assigned to a VLAN that provides limited services, or network access is not granted.

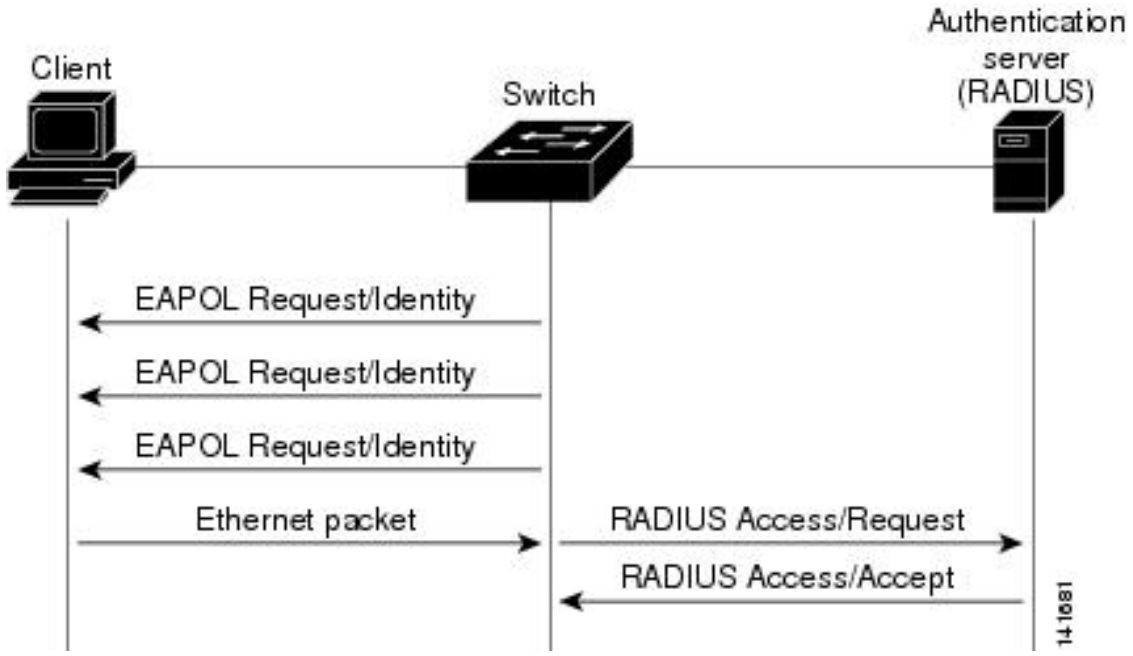
The specific exchange of EAP frames depends on the authentication method being used.

Figure 109: Message Exchange



If 802.1x authentication times out while waiting for an EAPOL message exchange and MAC authentication bypass is enabled, the switch can authorize the client when the switch detects an Ethernet packet from the client. The switch uses the MAC address of the client as its identity and includes this information in the RADIUS-access/request frame that is sent to the RADIUS server. After the server sends the switch the RADIUS-access/accept frame (authorization is successful), the port becomes authorized. If authorization fails and a guest VLAN is specified, the switch assigns the port to the guest VLAN. If the switch detects an EAPOL packet while waiting for an Ethernet packet, the switch stops the MAC authentication bypass process and starts 802.1x authentication.

Figure 110: Message Exchange During MAC Authentication Bypass



Port-Based Authentication Methods

Table 107: 802.1x Features

Authentication method	Mode			
	Single host	Multiple host	MDA	Multiple Authentication
802.1x	VLAN assignment Per-user ACL Filter-ID attribute Downloadable ACL Redirect URL	VLAN assignment	VLAN assignment Per-user ACL Filter-Id attribute Downloadable ACL Redirect URL	VLAN assignment Per-user ACL Filter-Id attribute Downloadable ACL Redirect URL
MAC authentication bypass	VLAN assignment Per-user ACL Filter-ID attribute Downloadable ACL Redirect URL	VLAN assignment	VLAN assignment Per-user ACL Filter-Id attribute Downloadable ACL Redirect URL	VLAN assignment Per-user ACL Filter-Id attribute Downloadable ACL Redirect URL
Standalone web authentication	Proxy ACL, Filter-Id attribute, downloadable ACL			

Authentication method	Mode			
	Single host	Multiple host	MDA	Multiple Authentication
NAC Layer 2 IP validation	Filter-Id attribute Downloadable ACL Redirect URL	Filter-Id attribute Downloadable ACL Redirect URL	Filter-Id attribute Downloadable ACL Redirect URL	Filter-Id attribute Downloadable ACL Redirect URL
Web authentication as fallback method	Proxy ACL Filter-Id attribute Downloadable ACL	Proxy ACL Filter-Id attribute Downloadable ACL	Proxy ACL Filter-Id attribute Downloadable ACL	Proxy ACL Filter-Id attribute Downloadable ACL

[9] For clients that do not support 802.1x authentication.

Per-User ACLs and Filter IDs

Note Using role-based ACLs as Filter ID is not recommended.

More than one host can be authenticated on MDA-enabled and multiauth ports. The ACL policy applied for one host does not effect the traffic of another host. If only one host is authenticated on a multi-host port, and the other hosts gain network access without authentication, the ACL policy for the first host can be applied to the other connected hosts by specifying any in the source address.

Ports in Authorized and Unauthorized States

During 802.1x authentication, depending on the switch port state, the switch can grant a client access to the network. The port starts in the *unauthorized* state. While in this state, the port that is not configured as a voice VLAN port disallows all ingress and egress traffic except for 802.1x authentication, Cisco Discovery Protocol, and STP packets. When a client is successfully authenticated, the port changes to the *authorized* state, allowing all traffic for the client to flow normally. If the port is configured as a voice VLAN port, the port allows VoIP traffic and 802.1x protocol packets before the client is successfully authenticated.

Note Cisco Discovery Protocol bypass is not supported and may cause a port to go into err-disabled state.

If a client that does not support 802.1x authentication connects to an unauthorized 802.1x port, the switch requests the client's identity. In this situation, the client does not respond to the request, the port remains in the unauthorized state, and the client is not granted access to the network.

In contrast, when an 802.1x-enabled client connects to a port that is not running the 802.1x standard, the client initiates the authentication process by sending the EAPOL-start frame. When no response is received, the client sends the request for a fixed number of times. Because no response is received, the client begins sending frames as if the port is in the authorized state.

You control the port authorization state by using the **authentication port-control** interface configuration command and these keywords:

- **force-authorized**: Disables 802.1x authentication and causes the port to change to the authorized state without any authentication exchange required. The port sends and receives normal traffic without 802.1x-based authentication of the client. This is the default setting.
- **force-unauthorized**: Causes the port to remain in the unauthorized state, ignoring all attempts by the client to authenticate. The switch cannot provide authentication services to the client through the port.
- **auto**: Enables 802.1x authentication and causes the port to begin in the unauthorized state, allowing only EAPOL frames to be sent and received through the port. The authentication process begins when the link state of the port changes from down to up or when an EAPOL-start frame is received. The switch requests the identity of the client and begins relaying authentication messages between the client and the authentication server. Each client attempting to access the network is uniquely identified by the switch by using the client MAC address.

If the client is successfully authenticated (receives an Accept frame from the authentication server), the port state changes to authorized, and all frames from the authenticated client are allowed through the port. If the authentication fails, the port remains in the unauthorized state, but authentication can be retried. If the authentication server cannot be reached, the switch can resend the request. If no response is received from the server after the specified number of attempts, authentication fails, and network access is not granted.

When a client logs off, it sends an EAPOL-logoff message, causing the switch port to change to the unauthorized state.

If the link state of a port changes from up to down, or if an EAPOL-logoff frame is received, the port returns to the unauthorized state.

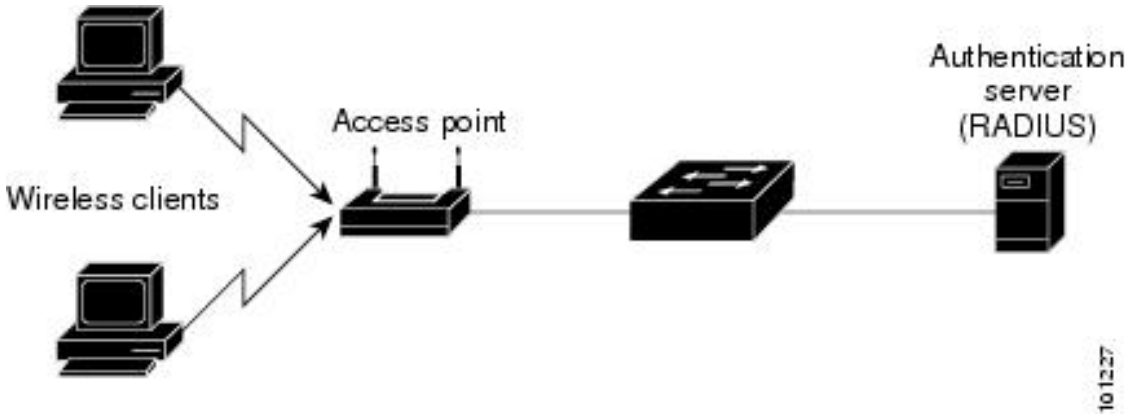
802.1x Host Mode

You can configure an 802.1x port for single-host or for multiple-hosts mode. In single-host mode, only one client can be connected to the 802.1x-enabled switch port. The switch detects the client by sending an EAPOL frame when the port link state changes to the up state. If a client leaves or is replaced with another client, the switch changes the port link state to down, and the port returns to the unauthorized state.

In multiple-hosts mode, you can attach multiple hosts to a single 802.1x-enabled port. In this mode, only one of the attached clients must be authorized for all clients to be granted network access. If the port becomes unauthorized (reauthentication fails or an EAPOL-logoff message is received), the switch denies network access to all of the attached clients.

In this topology, the wireless access point is responsible for authenticating the clients attached to it, and it also acts as a client to the switch.

Figure 111: Multiple Host Mode Example



Note For all host modes, the line protocol stays up before authorization when port-based authentication is configured.

The switch supports multidomain authentication (MDA), which allows both a data device and a voice device, such as an IP Phone, to connect to the same switch port.

Information About IEEE 802.1x Port-Based Authentication

The 802.1x standard defines a client-server-based access control and authentication protocol that prevents unauthorized clients from connecting to a LAN through publicly accessible ports unless they are properly authenticated. The authentication server authenticates each client connected to a switch port before making available any services offered by the switch or the LAN.

Note TACACS is not supported with 802.1x authentication.

Until the client is authenticated, 802.1x access control allows only Extensible Authentication Protocol over LAN (EAPOL), Cisco Discovery Protocol, and Spanning Tree Protocol (STP) traffic through the port to which the client is connected. After authentication is successful, normal traffic can pass through the port.

The table shown below lists the maximum number of session each client supports:

Client session	Maximum sessions supported
Maximum dot1x or MAB client sessions	2000
Maximum web-based authentication sessions	2000
Maximum dot1x sessions with critical-auth VLAN enabled and server re-initialized	2000
Maximum MAB sessions with various session features applied	2000
Maximum dot1x sessions with service templates or session features applied	2000

Access Session Limit Profile

An access session limit profile will allow you to limit the number of voice and data hosts connecting to a port. An access session limit profile will have higher priority compared to any host mode configuration. When an access session limit profile is configured the host mode configuration will be ignored.

You can create an access session limit profile by using the **access-session limit profile** command in the global configuration mode. You can configure the profile to limit the number of data and voice sessions allowed per interface. The profile can be configured to allow multiple hosts and to bypass authentication based on CDP packets if CDP bypass is supported.

The access session limit profile needs to be applied at an interface level.

You can also attach the access session limit profile to an interface template.

MAC Move

When a MAC address is authenticated on one switch port, that address is not allowed on another authentication manager-enabled port of the switch. If the switch detects that same MAC address on another authentication manager-enabled port, the address is not allowed.

There are situations where a MAC address might need to move from one port to another on the same switch. For example, when there is another device (for example a hub or an IP phone) between an authenticated host and a switch port, you might want to disconnect the host from the device and connect it directly to another port on the same switch.

You can globally enable MAC move so the device is reauthenticated on the new port. When a host moves to a second port, the session on the first port is deleted, and the host is reauthenticated on the new port. MAC move is supported on all host modes. (The authenticated host can move to any port on the switch, no matter which host mode is enabled on that port.) When a MAC address moves from one port to another, the switch terminates the authenticated session on the original port and initiates a new authentication sequence on the new port. The MAC move feature applies to both voice and data hosts.

Note In open authentication mode, a MAC address is immediately moved from the original port to the new port, with no requirement for authorization on the new port.

MAC Replace

The MAC Replace feature can be configured to address the violation that occurs when a host attempts to connect to a port where another host was previously authenticated.

Note This feature does not apply to ports in multi-auth mode, because violations are not triggered in that mode. It does not apply to ports in multiple host mode, because in that mode, only the first host requires authentication.

If you configure the **authentication violation** interface configuration command with the **replace** keyword, the authentication process on a port in multidomain mode is:

- A new MAC address is received on a port with an existing authenticated MAC address.
- The authentication manager replaces the MAC address of the current data host on the port with the new MAC address.

- The authentication manager initiates the authentication process for the new MAC address.
- If the authentication manager determines that the new host is a voice host, the original voice host is removed.

If a port is in open authentication mode, any new MAC address is immediately added to the MAC address table.

802.1x Accounting

The 802.1x standard defines how users are authorized and authenticated for network access but does not keep track of network usage. 802.1x accounting is disabled by default. You can enable 802.1x accounting to monitor this activity on 802.1x-enabled ports:

- User successfully authenticates.
- User logs off.
- Link-down occurs.
- Reauthentication successfully occurs.
- Reauthentication fails.

The switch does not log 802.1x accounting information. Instead, it sends this information to the RADIUS server, which must be configured to log accounting messages.

802.1x Accounting Attribute-Value Pairs

The information sent to the RADIUS server is represented in the form of Attribute-Value (AV) pairs. These AV pairs provide data for different applications. (For example, a billing application might require information that is in the Acct-Input-Octets or the Acct-Output-Octets attributes of a RADIUS packet.)

AV pairs are automatically sent by a switch that is configured for 802.1x accounting. Three types of RADIUS accounting packets are sent by a switch:

- START: Sent when a new user session starts
- INTERIM: Sent during an existing session for updates
- STOP: Sent when a session terminates

Table 108: Accounting AV Pairs

Attribute Number	AV Pair Name	START	INTERIM	STOP
Attribute[1]	User-Name	Always	Always	Always
Attribute[4]	NAS-IP-Address	Always	Always	Always
Attribute[5]	NAS-Port	Always	Always	Always
Attribute[8]	Framed-IP-Address	Never	Sometimes[10]	Sometimes
Attribute[30]	Called-Station-ID	Always	Always	Always
Attribute[31]	Calling-Station-ID	Always	Always	Always

Attribute Number	AV Pair Name	START	INTERIM	STOP
Attribute[40]	Acct-Status-Type	Always	Always	Always
Attribute[41]	Acct-Delay-Time	Always	Always	Always
Attribute[42]	Acct-Input-Octets	Never	Always	Always
Attribute[43]	Acct-Output-Octets	Never	Always	Always
Attribute[47]	Acct-Input-Packets	Never	Always	Always
Attribute[48]	Acct-Output-Packets	Never	Always	Always
Attribute[44]	Acct-Session-ID	Always	Always	Always
Attribute[45]	Acct-Authentic	Always	Always	Always
Attribute[46]	Acct-Session-Time	Never	Always	Always
Attribute[49]	Acct-Terminate-Cause	Never	Never	Always
Attribute[61]	NAS-Port-Type	Always	Always	Always

[10] The Framed-IP-Address AV pair is sent when a valid static IP address is configured or w when a Dynamic Host Control Protocol (DHCP) binding exists for the host in the DHCP snooping bindings table.

802.1x Readiness Check

The 802.1x readiness check monitors 802.1x activity on all the switch ports and displays information about the devices connected to the ports that support 802.1x. You can use this feature to determine if the devices connected to the switch ports are 802.1x-capable. You use an alternate authentication such as MAC authentication bypass or web authentication for the devices that do not support 802.1x functionality.

This feature only works if the supplicant on the client supports a query with the NOTIFY EAP notification packet. The client must respond within the 802.1x timeout value.

Switch-to-RADIUS Server Communication

RADIUS security servers are identified by their hostname or IP address, hostname and specific UDP port numbers, or IP address and specific UDP port numbers. The combination of the IP address and UDP port number creates a unique identifier, which enables RADIUS requests to be sent to multiple UDP ports on a server at the same IP address. If two different host entries on the same RADIUS server are configured for the same service, for example, authentication, the second host entry configured acts as the fail-over backup to the first one. The RADIUS host entries are tried in the order that they were configured.

IEEE 802.1x Authentication

The following sections provide information about IEEE 802.1x authentication.

802.1x Authentication

These are the 802.1x authentication configuration guidelines:

- You must enable SISF-Based device tracking to use 802.1x authentication. By default, SISF-Based device tracking is disabled on a switch.
- When 802.1x authentication is enabled, ports are authenticated before any other Layer 2 or Layer 3 features are enabled.
- If the VLAN to which an 802.1x-enabled port is assigned changes, this change is transparent and does not affect the switch. For example, this change occurs if a port is assigned to a RADIUS server-assigned VLAN and is then assigned to a different VLAN after reauthentication.

  If the VLAN to which an 802.1x port is assigned to shut down, disabled, or removed, the port becomes unauthorized. For example, the port is unauthorized after the access VLAN to which a port is assigned shuts down or is removed.
- The 802.1x protocol is supported on Layer 2 static-access ports, voice VLAN ports, and Layer 3 routed ports, but it is not supported on these port types:
  - Dynamic ports: A port in dynamic mode can negotiate with its neighbor to become a trunk port. If you try to enable 802.1x authentication on a dynamic port, an error message appears, and 802.1x authentication is not enabled. If you try to change the mode of an 802.1x-enabled port to dynamic, an error message appears, and the port mode is not changed.
  - EtherChannel port: Do not configure a port that is an active or a not-yet-active member of an EtherChannel as an 802.1x port. If you try to enable 802.1x authentication on an EtherChannel port, an error message appears, and 802.1x authentication is not enabled.
  - Switched Port Analyzer (SPAN) and Remote SPAN (RSPAN) destination ports: You can enable 802.1x authentication on a port that is a SPAN or RSPAN destination port. However, 802.1x authentication is disabled until the port is removed as a SPAN or RSPAN destination port. You can enable 802.1x authentication on a SPAN or RSPAN source port.
- Before globally enabling 802.1x authentication on a switch by entering the **dot1x system-auth-control** global configuration command, remove the EtherChannel configuration from the interfaces on which 802.1x authentication and EtherChannel are configured.
- Filtering of system messages related to 802.1x authentication is supported.

Note We recommend that you configure all the dependent 802.1x CLIs under the same interface or on the same template.

Port-Based Authentication Manager CLI Commands

The authentication-manager interface-configuration commands control all the authentication methods, such as 802.1x, MAC authentication bypass, and web authentication. The authentication manager commands determine the priority and order of authentication methods applied to a connected host.

The authentication manager commands control generic authentication features, such as host-mode, violation mode, and the authentication timer. Generic authentication commands include the **authentication host-mode**, **authentication violation**, and **authentication timer** interface configuration commands.

802.1x-specific commands begin with the **dot1x** keyword. For example, the **authentication port-control auto** interface configuration command enables authentication on an interface.

To disable dot1x on a switch, remove the configuration globally by using the no dot1x system-auth-control , and also remove it from all configured interfaces.

Note If 802.1x authentication is globally disabled, other authentication methods are still enabled on that port, such as web authentication.

The **authentication manager** commands provide the same functionality as earlier 802.1x commands.

When filtering out verbose system messages generated by the authentication manager, the filtered content typically relates to authentication success. You can also filter verbose messages for 802.1x authentication and MAB authentication. There is a separate command for each authentication method:

- The **no authentication logging verbose** global configuration command filters verbose messages from the authentication manager.
- The **no dot1x logging verbose** global configuration command filters 802.1x authentication verbose messages.
- The **no mab logging verbose** global configuration command filters MAC authentication bypass (MAB) verbose messages

Default 802.1x Authentication Configuration

Table 109: Default 802.1x Authentication Configuration

Feature	Default Setting
Switch 802.1x enable state	Disabled.
Per-port 802.1x enable state	Disabled (force-authorized). The port sends and receives normal traffic without 802.1x-based authentication of the client.
AAA	Disabled.
RADIUS server • IP address • UDP authentication port • Default accounting port • Key	• None specified. • 1645. • 1646. • None specified.
Host mode	Single-host mode.
Control direction	Bidirectional control.
Periodic reauthentication	Disabled.
Number of seconds between reauthentication attempts	3600 seconds.

Feature	Default Setting
Re-authentication number	Twice (number of times that the switch restarts the authentication process before the port changes to the unauthorized state).
Quiet period	60 seconds (number of seconds that the switch remains in the quiet state following a failed authentication exchange with the client).
Retransmission time	30 seconds (number of seconds that the switch should wait for a response to an EAP request/identity frame from the client before resending the request).
Maximum retransmission number	2 times (number of times that the switch will send an EAP-request/identity frame before restarting the authentication process).
Client timeout period	30 seconds (when relaying a request from the authentication server to the client, the amount of time the switch waits for a response before resending the request to the client.)
Authentication server timeout period	30 seconds (when relaying a response from the client to the authentication server, the amount of time the switch waits for a reply before resending the response to the server.) You can change this timeout period by using the dot1x timeout server-timeout interface configuration command.
Inactivity timeout	Disabled.
Guest VLAN	None specified.
Inaccessible authentication bypass	Disabled.
Restricted VLAN	None specified.
Authenticator (switch) mode	None specified.
MAC authentication bypass	Disabled.
Voice-aware security	Disabled.

802.1x Authentication with VLAN Assignment

The switch supports 802.1x authentication with VLAN assignment. After successful 802.1x authentication of a port, the RADIUS server sends the VLAN assignment to configure the switch port. The RADIUS server database maintains the username-to-VLAN mappings, assigning the VLAN based on the username of the client connected to the switch port. You can use this feature to limit network access for certain users.

When a voice device is authorized and the RADIUS server returned an authorized VLAN, the voice VLAN on the port is configured to send and receive packets on the assigned voice VLAN. Voice VLAN assignment behaves the same as data VLAN assignment on multidomain authentication (MDA)-enabled ports.

When configured on the switch and the RADIUS server, 802.1x authentication with VLAN assignment has these characteristics:

- If no VLAN is supplied by the RADIUS server or if 802.1x authentication is disabled, the port is configured in its access VLAN after successful authentication. Recall that an access VLAN is a VLAN assigned to an access port. All packets sent from or received on this port belong to this VLAN.
- If 802.1x authentication is enabled but the VLAN information from the RADIUS server is not valid, authorization fails and configured VLAN remains in use. This prevents ports from appearing unexpectedly in an inappropriate VLAN because of a configuration error.

  Configuration errors could include specifying a VLAN for a routed port, a malformed VLAN ID, a nonexistent or internal (routed port) VLAN ID, an RSPAN VLAN, a shut down or suspended VLAN. In the case of a multidomain host port, configuration errors can also be due to an attempted assignment of a data VLAN that matches the configured or assigned voice VLAN ID (or the reverse).
- If 802.1x authentication is enabled and all information from the RADIUS server is valid, the authorized device is placed in the specified VLAN after authentication.
- If the multiple-hosts mode is enabled on an 802.1x port, all hosts are placed in the same VLAN (specified by the RADIUS server) as the first authenticated host.
- Enabling port security does not impact the RADIUS server-assigned VLAN behavior.
- If 802.1x authentication is disabled on the port, it is returned to the configured access VLAN and configured voice VLAN.
- If an 802.1x port is authenticated and put in the RADIUS server-assigned VLAN, any change to the port access VLAN configuration does not take effect. In the case of a multidomain host, the same applies to voice devices when the port is fully authorized with these exceptions:
  - If the VLAN configuration change of one device results in matching the other device configured or assigned VLAN, then authorization of all devices on the port is terminated and multidomain host mode is disabled until a valid configuration is restored where data and voice device configured VLANs no longer match.
  - If a voice device is authorized and is using a downloaded voice VLAN, the removal of the voice VLAN configuration, or modifying the configuration value to *dot1p* or *untagged* results in voice device unauthorization and the disablement of multi-domain host mode.

When the port is in the force authorized, force unauthorized, unauthorized, or shutdown state, it is put into the configured access VLAN.

To configure VLAN assignment you need to perform these tasks:

- Enable AAA authorization by using the **network** keyword to allow interface configuration from the RADIUS server.
- Enable 802.1x authentication. (The VLAN assignment feature is automatically enabled when you configure 802.1x authentication on an access port).
- Assign vendor-specific tunnel attributes in the RADIUS server. The RADIUS server must return these attributes to the switch:

- [64] Tunnel-Type = VLAN
- [65] Tunnel-Medium-Type = 802
- [81] Tunnel-Private-Group-ID = VLAN name or VLAN ID
- [83] Tunnel-Preference

Attribute [64] must contain the value *VLAN* (type 13). Attribute [65] must contain the value *802* (type 6). Attribute [81] specifies the *VLAN name* or *VLAN ID* assigned to the IEEE 802.1x-authenticated user.

802.1x Authentication with Per-User ACLs

You can enable per-user access control lists (ACLs) to provide different levels of network access and service to an 802.1x-authenticated user. When the RADIUS server authenticates a user connected to an 802.1x port, it retrieves the ACL attributes based on the user identity and sends them to the switch. The switch applies the attributes to the 802.1x port for the duration of the user session. The switch removes the per-user ACL configuration when the session is over, if authentication fails, or if a link-down condition occurs. The switch does not save RADIUS-specified ACLs in the running configuration. When the port is unauthorized, the switch removes the ACL from the port.

You can configure router ACLs and input port ACLs on the same switch. However, a port ACL takes precedence over a router ACL. If you apply input port ACL to an interface that belongs to a VLAN, the port ACL takes precedence over an input router ACL applied to the VLAN interface. Incoming packets received on the port, to which a port ACL is applied, are filtered by the port ACL. Incoming routed packets received on other ports are filtered by the router ACL. Outgoing routed packets are filtered by the router ACL. To avoid configuration conflicts, you should carefully plan the user profiles stored on the RADIUS server.

RADIUS supports per-user attributes, including vendor-specific attributes. These vendor-specific attributes (VSAs) are in octet-string format and are passed to the switch during the authentication process. The VSAs used for per-user ACLs are inacl#<*n*> for the ingress direction and outacl#<*n*> for the egress direction. MAC ACLs are supported only in the ingress direction. The switch supports VSAs only in the ingress direction. It does not support port ACLs in the egress direction on Layer 2 ports.

Use only the extended ACL syntax style to define the per-user configuration stored on the RADIUS server. When the definitions are passed from the RADIUS server, they are created by using the extended naming convention. However, if you use the Filter-Id attribute, it can point to a standard ACL.

You can use the Filter-Id attribute to specify an inbound or outbound ACL that is already configured on the switch. The attribute contains the ACL number followed by *.in* for ingress filtering or *.out* for egress filtering. If the RADIUS server does not allow the *.in* or *.out* syntax, the access list is applied to the outbound ACL by default. The user is marked unauthorized if the Filter-Id sent from the RADIUS server is not configured on the device. The Filter-Id attribute is supported only for IP ACLs numbered in the range of 1 to 199 (IP standard ACLs) and 1300 to 2699 (IP extended ACLs).

The maximum size of the per-user ACL is 4000 ASCII characters but is limited by the maximum size of RADIUS-server per-user ACLs.

You must meet the following prerequisites to configure per-user ACLs:

- Enable AAA authentication.
- Enable AAA authorization by using the **network** keyword to allow interface configuration from the RADIUS server.
- Enable 802.1x authentication.

- Configure the user profile and VSAs on the RADIUS server.
- Configure the 802.1x port for single-host mode.

Note Per-user ACLs are supported only in single-host mode.

802.1x Authentication with Downloadable ACLs and Redirect URLs

You can download ACLs and redirect URLs from a RADIUS server to the switch during 802.1x authentication or MAC authentication bypass of the host. You can also download ACLs during web authentication. A downloadable ACL is also referred to as a *dACL*.

If more than one host is authenticated and the host is in single-host, MDA, or multiple-authentication mode, the switch changes the source address of the ACL to the host IP address.

You can apply the ACLs and redirect URLs to all the devices connected to the 802.1x-enabled port.

If no ACLs are downloaded during 802.1x authentication, the switch applies the static default ACL on the port to the host. On a voice VLAN port configured in multi-auth or MDA mode, the switch applies the ACL only to the phone as part of the authorization policies.

Multiple dACLs of the same type (IPv4 or IPv6) are not supported through Cisco Identity Services Engine (ISE). Ensure that only unique DACLs are sent from Cisco ISE.

The 802.1x and MAB authentication methods support two authentication modes, *open* and *closed*. If there is no static ACL on a port in *closed* authentication mode:

- When the first host authenticates, the authorization policy is applied without IP address insertion.
- When a second host is detected, the policies for the first host are refreshed, and policies for the first and subsequent sessions are enforced with IP address insertion.

If there is no static ACL on a port in *open* authentication mode:

- Policies are enforced with IP address insertion to prevent security breaches.

To control access for hosts with no authorization policy, you can configure a directive. The supported values for the directive are *open* and *default*. When you configure the *open* directive, all traffic is allowed. The *default* directive subjects traffic to the access provided by the port. You can configure the directive either in the user profile on the AAA server or on the switch. To configure the directive on the AAA server, use the **authz-directive =<open/default>** global command. To configure the directive on the switch, use the **epm access-control open** global configuration command.

Note The default value of the directive is *default.*

For a URL redirect ACL:

- Packets that match a permit access control entry (ACE) rule are sent to the CPU for forwarding to the AAA server.
- Packets that match a deny ACE rule are forwarded through the switch.

- Packets that match neither the permit ACE rule or deny ACE rule are processed by the next dACL, and if there is no dACL, the packets hit the implicit-deny ACL and are dropped.

VLAN ID-Based MAC Authentication

You can use VLAN ID-based MAC authentication if you wish to authenticate hosts based on a static VLAN ID instead of a downloadable VLAN. When you have a static VLAN policy configured on your switch, VLAN information is sent to an IAS (Microsoft) RADIUS server along with the MAC address of each host for authentication. The VLAN ID configured on the connected port is used for MAC authentication. By using VLAN ID-based MAC authentication with an IAS server, you can have a fixed number of VLANs in the network.

The feature also limits the number of VLANs monitored and handled by STP. The network can be managed as a fixed VLAN.

IEEE 802.1x Authentication with MAC Authentication Bypass

You can configure the switch to authorize clients based on the client MAC address by using the MAC authentication bypass feature. For example, you can enable this feature on IEEE 802.1x ports connected to devices such as printers.

If IEEE 802.1x authentication times out while waiting for an EAPOL response from the client, the switch tries to authorize the client by using MAC authentication bypass.

When the MAC authentication bypass feature is enabled on an IEEE 802.1x port, the switch uses the MAC address as the client identity. The authentication server has a database of client MAC addresses that are allowed network access. After detecting a client on an IEEE 802.1x port, the switch waits for an Ethernet packet from the client. The switch sends the authentication server a RADIUS-access/request frame with a username and password based on the MAC address. If authorization succeeds, the switch grants the client access to the network. If authorization fails, the switch assigns the port to the guest VLAN if one is configured. This process works for most client devices; however, it does not work for clients that use an alternate MAC address format. You can configure how MAB authentication is performed for clients with MAC addresses that deviate from the standard format or where the RADIUS configuration requires the user name and password to differ.

If an EAPOL packet is detected on the interface during the lifetime of the link, the switch determines that the device connected to that interface is an 802.1x-capable supplicant and uses 802.1x authentication (not MAC authentication bypass) to authorize the interface. EAPOL history is cleared if the interface link status goes down.

If the switch already authorized a port by using MAC authentication bypass and detects an IEEE 802.1x supplicant, the switch does not unauthorize the client connected to the port. When reauthentication occurs, the switch uses the authentication or reauthentication methods configured on the port, if the previous session ended because the Termination-Action RADIUS attribute value is DEFAULT.

Clients that were authorized with MAC authentication bypass can be reauthenticated. The reauthentication process is the same as that for clients that were authenticated with IEEE 802.1x. During reauthentication, the port remains in the previously assigned VLAN. If reauthentication is successful, the switch keeps the port in the same VLAN. If reauthentication fails, the switch assigns the port to the guest VLAN, if one is configured.

If reauthentication is based on the Session-Timeout RADIUS attribute (Attribute[27]) and the Termination-Action RADIUS attribute (Attribute [29]) and if the Termination-Action RADIUS attribute (Attribute [29]) action is *Initialize* (the attribute value is *DEFAULT*), the MAC authentication bypass session ends, and connectivity is lost during reauthentication. If MAC authentication bypass is enabled and the IEEE 802.1x authentication times out, the switch uses the MAC authentication bypass feature to initiate

re-authorization. For more information about these AV pairs, see RFC 3580, "IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines."

MAC authentication bypass interacts with the features:

- IEEE 802.1x authentication: You can enable MAC authentication bypass only if 802.1x authentication is enabled on the port .
- Guest VLAN: If a client has an invalid MAC address identity, the switch assigns the client to a guest VLAN if one is configured.
- Restricted VLAN: This feature is not supported when the client connected to an IEEE 802.lx port is authenticated with MAC authentication bypass.
- Port security
- Voice VLAN
- Private VLAN: You can assign a client to a private VLAN.
- Network Edge Access Topology (NEAT): MAB and NEAT are mutually exclusive. You cannot enable MAB when NEAT is enabled on an interface, and you should not enable NEAT when MAB is enabled on an interface.

MAC Authentication Bypass Configuration Guidelines

These are the MAC authentication bypass configuration guidelines:

- Unless otherwise stated, the MAC authentication bypass guidelines are the same as the 802.1x authentication guidelines.
- If you disable MAC authentication bypass from a port after the port has been authorized with its MAC address, the port state is not affected.
- If the port is in the unauthorized state and the client MAC address is not the authentication-server database, the port remains in the unauthorized state. However, if the client MAC address is added to the database, the switch can use MAC authentication bypass to re-authorize the port.
- If the port is in the authorized state, the port remains in this state until re-authorization occurs.
- You can configure a timeout period for hosts that are connected by MAC authentication bypass but are inactive. The range is 1to 65535 seconds.

802.1x Multiple Authentication Mode

Multiple-authentication (multiauth) mode allows multiple authenticated clients on the data VLAN and voice VLAN. Each host is individually authenticated. There is no limit to the number of data or voice device that can be authenticated on a multiauthport.

If a hub or access point is connected to an 802.1x-enabled port, each connected client must be authenticated. For non-802.1x devices, you can use MAC authentication bypass or web authentication as the per-host authentication fallback method to authenticate different hosts with different methods on a single port.

You can assign a RADIUS-server-supplied VLAN in multi-auth mode, under the following conditions:

- The host is the first host authorized on the port, and the RADIUS server supplies VLAN information
- Subsequent hosts are authorized with a VLAN that matches the operational VLAN.

- A host is authorized on the port with no VLAN assignment, and subsequent hosts either have no VLAN assignment, or their VLAN information matches the operational VLAN.
- The first host authorized on the port has a group VLAN assignment, and subsequent hosts either have no VLAN assignment, or their group VLAN matches the group VLAN on the port. Subsequent hosts must use the same VLAN from the VLAN group as the first host. If a VLAN list is used, all hosts are subject to the conditions specified in the VLAN list.
- After a VLAN is assigned to a host on the port, subsequent hosts must have matching VLAN information or be denied access to the port.
- The behavior of the critical-auth VLAN is not changed for multi-auth mode. When a host tries to authenticate and the server is not reachable, all authorized hosts are reinitialized in the configured VLAN.

Multi-auth Per User VLAN assignment

The Multi-auth Per User VLAN assignment feature allows you to create multiple operational access VLANs based on VLANs assigned to the clients on the port that has a single configured access VLAN. The port configured as an access port where the traffic for all the VLANs associated with data domain is not dot1q tagged, and these VLANs are treated as native VLANs.

The number of hosts per multi-auth port is 8, however there can be more hosts.

The following scenarios are associated with the multi-auth Per User VLAN assignments:

Scenario One

When a hub is connected to an access port, and the port is configured with an access VLAN (V0).

The host (H1) is assigned to VLAN (V1) through the hub. The operational VLAN of the port is changed to V1. This behaviour is similar on a single-host or multi-domain-auth port.

When a second host (H2) is connected and gets assigned to VLAN (V2), the port will have two operational VLANs (V1 and V2). If H1 and H2 sends untagged ingress traffic, H1 traffic is mapped to VLAN (V1) and H2 traffic to VLAN (V2), all egress traffic going out of the port on VLAN (V1) and VLAN (V2) are untagged.

If both the hosts, H1 and H2 are logged out or the sessions are removed due to some reason then VLAN (V1) and VLAN (V2) are removed from the port, and the configured VLAN (V0) is restored on the port.

Scenario Two

When a hub is connected to an access port, and the port is configured with an access VLAN (V0). The host (H1) is assigned to VLAN (V1) through the hub. The operational VLAN of the port is changed to V1.

When a second host (H2) is connected and gets authorized without explicit vlan policy, H2 is expected to use the configured VLAN (V0) that is restored on the port. A ll egress traffic going out of two operational VLANs, VLAN (V0) and VLAN (V1) are untagged.

If host (H2) is logged out or the session is removed due to some reason then the configured VLAN (V0) is removed from the port, and VLAN (V1) becomes the only operational VLAN on the port.

Scenario Three

When a hub is connected to an access port in open mode, and the port is configured with an access VLAN (V0) .

The host (H1) is assigned to VLAN (V1) through the hub. The operational VLAN of the port is changed to V1. When a second host (H2) is connected and remains unauthorized, it still has access to operational VLAN (V1) due to open mode.

If host H1 is logged out or the session is removed due to some reason, VLAN (V1) is removed from the port and host (H2) gets assigned to VLAN (V0).

Note The combination of Open mode and VLAN assignment has an adverse affect on host (H2) because it has an IP address in the subnet that corresponds to VLAN (V1).

Limitation in Multi-Auth Per User VLAN assignment

In the Multi-Auth Per User VLAN Assignment feature, egress traffic from multiple VLANs are untagged on a port where the hosts receive traffic that is not meant for them. This can be a problem with broadcast and multicast traffic.

- **IPv4 ARPs**: Hosts receive ARP packets from other subnets. This is a problem if two subnets in different Virtual Routing and Forwarding (VRF) tables with overlapping IP address range are active on the port. The host ARP cache may get invalid entries.
- **IPv6 Control Packets**: In IPv6 deployments, Router Advertisements (RA) are processed by hosts that are not supposed to receive them. When a host from one VLAN receives RA from a different VLAN, the host assign incorrect IPv6 address to itself. Such a host is unable to get access to the network.

  The workaround is to enable the IPv6 first hop security so that the broadcast ICMPv6 packets are converted to unicast and sent out from multi-auth enabled ports.. The packet is replicated for each client in multi-auth port belonging to the VLAN and the destination MAC is set to an individual client. Ports having one VLAN, ICMPv6 packets broadcast normally.
- **IP Multicast**: Multicast traffic destined to a multicast group gets replicated for different VLANs if the hosts on those VLANs join the multicast group. When two hosts in different VLANs join a multicast group (on the same mutli-auth port), two copies of each multicast packet are sent out from that port.

802.1x Authentication with Guest VLAN

You can configure a guest VLAN for each 802.1x port on the switch to provide limited services to clients, such as downloading the 802.1x client. These clients might be upgrading their system for 802.1x authentication, and some hosts, such as Windows 98 systems, might not be IEEE 802.1x-capable.

When you enable a guest VLAN on an 802.1x port, the switch assigns clients to a guest VLAN when the switch does not receive a response to its EAP request/identity frame or when EAPOL packets are not sent by the client.

The switch maintains the EAPOL packet history. If an EAPOL packet is detected on the interface during the lifetime of the link, the switch determines that the device connected to that interface is an IEEE 802.1x-capable supplicant, and the interface does not change to the guest VLAN state. EAPOL history is cleared if the interface link status goes down. If no EAPOL packet is detected on the interface, the interface changes to the guest VLAN state.

If the switch is trying to authorize an 802.1x-capable voice device and the AAA server is unavailable, the authorization attempt fails, but the detection of the EAPOL packet is saved in the EAPOL history. When the AAA server becomes available, the switch authorizes the voice device. However, the switch no longer allows other devices access to the guest VLAN. To prevent this situation, use one of these command sequences:

- Enter the **authentication event no-response action authorize vlan** *vlan-id* interface configuration command to allow access to the guest VLAN.

- Enter the **shutdown** interface configuration command followed by the **no shutdown** interface configuration command to restart the port.

If devices send EAPOL packets to the switch during the lifetime of the link, the switch no longer allows clients that fail authentication access to the guest VLAN.

Note If an EAPOL packet is detected after the interface has changed to the guest VLAN, the interface reverts to an unauthorized state, and 802.1x authentication restarts.

Any number of 802.1x-incapable clients are allowed access when the switch port is moved to the guest VLAN. If an 802.1x-capable client joins the same port on which the guest VLAN is configured, the port is put into the unauthorized state in the user-configured access VLAN, and authentication is restarted.

Guest VLANs are supported on 802.1x ports in single host, multiple host, multi-auth and multi-domain modes.

You can configure any active VLAN except an RSPAN VLAN, a private VLAN, or a voice VLAN as an 802.1x guest VLAN. The guest VLAN feature is not supported on internal VLANs (routed ports) or trunk ports; it is supported only on access ports.

The switch supports *MAC authentication bypass*. When MAC authentication bypass is enabled on an 802.1x port, the switch can authorize clients based on the client MAC address when IEEE 802.1x authentication times out while waiting for an EAPOL message exchange. After detecting a client on an 802.1x port, the switch waits for an Ethernet packet from the client. The switch sends the authentication server a RADIUS-access/request frame with a username and password based on the MAC address. If authorization succeeds, the switch grants the client access to the network. If authorization fails, the switch assigns the port to the guest VLAN if one is specified.

802.1x Authentication with Restricted VLAN

You can configure a restricted VLAN (also referred to as an *authentication failed VLAN*) for each IEEE 802.1x port on a switch to provide limited services to clients that cannot access the guest VLAN. These clients are 802.1x-compliant and cannot access another VLAN because they fail the authentication process. A restricted VLAN allows users without valid credentials in an authentication server (typically, visitors to an enterprise) to access a limited set of services. The administrator can control the services available to the restricted VLAN.

Note You can configure a VLAN to be both the guest VLAN and the restricted VLAN if you want to provide the same services to both types of users.

Without this feature, the client attempts and fails authentication indefinitely, and the switch port remains in the spanning-tree blocking state. With this feature, you can configure the switch port to be in the restricted VLAN after a specified number of authentication attempts (the default value is 3 attempts).

The authenticator counts the failed authentication attempts for the client. When this count exceeds the configured maximum number of authentication attempts, the port moves to the restricted VLAN. The failed attempt count increments when the RADIUS server replies with either an *EAP failure* or an empty response without an EAP packet. When the port moves into the restricted VLAN, the failed attempt counter resets.

Users who fail authentication remain in the restricted VLAN until the next reauthentication attempt. A port in the restricted VLAN tries to reauthenticate at configured intervals (the default is 60 seconds). If reauthentication fails, the port remains in the restricted VLAN. If reauthentication is successful, the port moves either to the configured VLAN or to a VLAN sent by the RADIUS server. You can disable reauthentication.

If you do this, the only way to restart the authentication process is for the port to receive a *link down* or *EAP logoff* event. We recommend that you keep reauthentication enabled if a client might connect through a hub. When a client disconnects from the hub, the port might not receive the *link down* or *EAP logoff* event.

After a port moves to the restricted VLAN, a simulated EAP success message is sent to the client. This prevents clients from indefinitely attempting authentication. Some clients (for example, devices running Windows XP) cannot implement DHCP without EAP success.

Restricted VLANs are supported on 802.1x ports in all host modes and on Layer 2 ports.

You can configure any active VLAN except an RSPAN VLAN, a primary private VLAN, or a voice VLAN as an 802.1x restricted VLAN. The restricted VLAN feature is not supported on internal VLANs (routed ports) or trunk ports; it is supported only on access ports.

Other security port features such as dynamic ARP Inspection, DHCP snooping, and IP source guard can be configured independently on a restricted VLAN.

802.1x Authentication with Inaccessible Authentication Bypass

Use the inaccessible authentication bypass feature, also referred to as *critical authentication* or the *AAA fail policy*, when the switch cannot reach the configured RADIUS servers and new hosts cannot be authenticated. You can configure the switch to connect those hosts to *critical ports.*

When a new host tries to connect to the critical port, that host is moved to a user-specified access VLAN, the *critical VLAN*. The administrator gives limited authentication to the hosts.

When the switch tries to authenticate a host connected to a critical port, the switch checks the status of the configured RADIUS server. If a server is available, the switch can authenticate the host. However, if all the RADIUS servers are unavailable, the switch grants network access to the host and puts the port in the *critical-authentication* state, which is a special case of the authentication state.

Note If *critical authentication* is configured on interface, then vlan used for critical authorization (*critical vlan*) should be active on the switch. If the *critical vlan* is inactive (or) down, *critical authentication* session will keep trying to enable inactive vlan and fail repeatedly. This can lead to large amount of memory holding.

Inaccessible Authentication Bypass Support on Multiple-Authentication Ports

When a port is configured on any host mode and the AAA server is unavailable, the port is then configured to multi-host mode and moved to the critical VLAN. To support this inaccessible bypass on multiple-authentication (multiauth) ports, use the **authentication event server dead action reinitialize vlan** *vlan-id* command. When a new host tries to connect to the critical port, that port is reinitialized and all the connected hosts are moved to the user-specified access VLAN.

This command is supported on all host modes.

Inaccessible Authentication Bypass Authentication Results

The behavior of the inaccessible authentication bypass feature depends on the authorization state of the port:

- If the port is unauthorized when a host connected to a critical port tries to authenticate and all servers are unavailable, the switch puts the port in the critical-authentication state in the RADIUS-configured or user-specified access VLAN.

- If the port is already authorized and reauthentication occurs, the switch puts the critical port in the critical-authentication state in the current VLAN, which might be the one previously assigned by the RADIUS server.
- If the RADIUS server becomes unavailable during an authentication exchange, the current exchange times out, and the switch puts the critical port in the critical-authentication state during the next authentication attempt.

You can configure the critical port to reinitialize hosts and move them out of the critical VLAN when the RADIUS server is again available. When this is configured, all critical ports in the critical-authentication state are automatically reauthenticated.

Inaccessible Authentication Bypass Feature Interactions

Inaccessible authentication bypass interacts with these features:

- Guest VLAN: Inaccessible authentication bypass is compatible with guest VLAN. When a guest VLAN is enabled on 8021.x port, the features interact as follows:
  - If at least one RADIUS server is available, the switch assigns a client to a guest VLAN when the switch does not receive a response to its EAP request/identity frame or when EAPOL packets are not sent by the client.
  - If all the RADIUS servers are not available and the client is connected to a critical port, the switch authenticates the client and puts the critical port in the critical-authentication state in the RADIUS-configured or user-specified access VLAN.
  - If all the RADIUS servers are not available and the client is not connected to a critical port, the switch might not assign clients to the guest VLAN if one is configured.
  - If all the RADIUS servers are not available and if a client is connected to a critical port and was previously assigned to a guest VLAN, the switch keeps the port in the guest VLAN.
- Restricted VLAN: If the port is already authorized in a restricted VLAN and the RADIUS servers are unavailable, the switch puts the critical port in the critical-authentication state in the restricted VLAN.
- 802.1x accounting: Accounting is not affected if the RADIUS servers are unavailable.
- Private VLAN: You can configure inaccessible authentication bypass on a private VLAN host port. The access VLAN must be a secondary private VLAN.
- Voice VLAN: Inaccessible authentication bypass is compatible with voice VLAN, but the RADIUS-configured or user-specified access VLAN and the voice VLAN must be different.
- Remote Switched Port Analyzer (RSPAN): Do not configure an RSPAN VLAN as the RADIUS-configured or user-specified access VLAN for inaccessible authentication bypass.

VLAN Assignment, Guest VLAN, Restricted VLAN, and Inaccessible Authentication Bypass

These are the configuration guidelines for VLAN assignment, guest VLAN, restricted VLAN, and inaccessible authentication bypass:

- When 802.1x authentication is enabled on a port, you cannot configure a port VLAN that is equal to a voice VLAN.

- You can configure any VLAN except an RSPAN VLAN or a voice VLAN as an 802.1x guest VLAN. The guest VLAN feature is not supported on internal VLANs (routed ports) or trunk ports; it is supported only on access ports.
- After you configure a guest VLAN for an 802.1x port to which a DHCP client is connected, you might need to get a host IP address from a DHCP server. You can change the settings for restarting the 802.1x authentication process on the switch before the DHCP process on the client times out and tries to get a host IP address from the DHCP server. Decrease the settings for the 802.1x authentication process (**authentication timer inactivity** and **authentication timer reauthentication** interface configuration commands). The amount to decrease the settings depends on the connected 802.1x client type.
- When configuring the inaccessible authentication bypass feature, follow these guidelines:
  - The feature is supported on 802.1x port in single-host mode and multihosts mode.
  - If the client is running Windows XP and the port to which the client is connected is in the critical-authentication state, Windows XP might report that the interface is not authenticated.
  - If the Windows XP client is configured for DHCP and has an IP address from the DHCP server, receiving an EAP-Success message on a critical port might not re-initiate the DHCP configuration process.
  - You can configure the inaccessible authentication bypass feature and the restricted VLAN on an 802.1x port. If the switch tries to reauthenticate a critical port in a restricted VLAN and all the RADIUS servers are unavailable, switch changes the port state to the critical authentication state and remains in the restricted VLAN.
- You can configure any VLAN except an RSPAN VLAN or a voice VLAN as an 802.1x restricted VLAN. The restricted VLAN feature is not supported on internal VLANs (routed ports) or trunk ports; it is supported only on access ports.

802.1x Critical Voice VLAN

When an IP phone connected to a port is authenticated by the Cisco Identity Services Engine (ISE), the phone is put into the voice domain. If the ISE is not reachable, the switch cannot determine if the device is a voice device. If the server is unavailable, the phone cannot access the voice network and therefore cannot operate.

For data traffic, you can configure inaccessible authentication bypass, or critical authentication, to allow traffic to pass through on the native VLAN when the server is not available. If the RADIUS authentication server is unavailable (down) and inaccessible authentication bypass is enabled, the switch grants the client access to the network and puts the port in the critical-authentication state in the RADIUS-configured or the user-specified access VLAN. When the switch cannot reach the configured RADIUS servers and new hosts cannot be authenticated, the switch connects those hosts to critical ports. A new host trying to connect to the critical port is moved to a user-specified access VLAN, the critical VLAN, and granted limited authentication.

Note Dynamic assignment of critical voice VLAN is not supported with nested service templates. It causes the device to switch between VLANs continuously in a loop.

You can enter the **authentication event server dead action authorize voice** interface configuration command to configure the critical voice VLAN feature. When the ISE does not respond, the port goes into critical authentication mode. When traffic coming from the host is tagged with the voice VLAN, the connected device

(the phone) is put in the configured voice VLAN for the port. The IP phones learn the voice VLAN identification through Cisco Discovery Protocol (Cisco devices) or through LLDP or DHCP.

You can configure the voice VLAN for a port by entering the **switchport voice vlan** *vlan-id* interface configuration command.

This feature is supported in multidomain and multi-auth host modes. Although you can enter the command when the switch in single-host or multi-host mode, the command has no effect unless the device changes to multidomain or multi-auth host mode.

IEEE 802.1x Authentication with Voice VLAN Ports

A voice VLAN port is a special access port associated with two VLAN identifiers:

- VVID to carry voice traffic to and from the IP phone. The VVID is used to configure the IP phone connected to the port.
- PVID to carry the data traffic to and from the workstation connected to the switch through the IP phone. The PVID is the native VLAN of the port.

The IP phone uses the VVID for its voice traffic, regardless of the authorization state of the port. This allows the phone to work independently of IEEE 802.1x authentication.

In single-host mode, only the IP phone is allowed on the voice VLAN. In multiple-hosts mode, additional clients can send traffic on the voice VLAN after a supplicant is authenticated on the PVID. When multiple-hosts mode is enabled, the supplicant authentication affects both the PVID and the VVID.

A voice VLAN port becomes active when there is a link, and the device MAC address appears after the first Cisco Discovery Protocol message from the IP phone. Cisco IP phones do not relay Cisco Discovery Protocol messages from other devices. As a result, if several IP phones are connected in series, the switch recognizes only the one directly connected to it. When IEEE 802.1x authentication is enabled on a voice VLAN port, the switch drops packets from unrecognized IP phones more than one hop away.

When IEEE 802.1x authentication is enabled on a switch port, you can configure an access port VLAN that is also a voice VLAN.

When IP phones are connected to an 802.1x-enabled switch port that is in single host mode, the switch grants the phones network access without authenticating them. We recommend that you use multidomain authentication (MDA) on the port to authenticate both a data device and a voice device, such as an IP phone

Note If you enable IEEE 802.1x authentication on an access port on which a voice VLAN is configured and to which a Cisco IP Phone is connected, the Cisco IP phone loses connectivity to the switch for up to 30 seconds.

Information About IEEE 802.1x Port-Based Authentication

The 802.1x standard defines a client-server-based access control and authentication protocol that prevents unauthorized clients from connecting to a LAN through publicly accessible ports unless they are properly authenticated. The authentication server authenticates each client connected to a switch port before making available any services offered by the switch or the LAN.

Note TACACS is not supported with 802.1x authentication.

Until the client is authenticated, 802.1x access control allows only Extensible Authentication Protocol over LAN (EAPOL), Cisco Discovery Protocol, and Spanning Tree Protocol (STP) traffic through the port to which the client is connected. After authentication is successful, normal traffic can pass through the port.

The table shown below lists the maximum number of session each client supports:

Client session	Maximum sessions supported
Maximum dot1x or MAB client sessions	2000
Maximum web-based authentication sessions	2000
Maximum dot1x sessions with critical-auth VLAN enabled and server re-initialized	2000
Maximum MAB sessions with various session features applied	2000
Maximum dot1x sessions with service templates or session features applied	2000

Flexible Authentication Ordering

You can use flexible authentication ordering to configure the order of methods that a port uses to authenticate a new host. The IEEE 802.1X Flexible Authentication feature supports three authentication methods:

- dot1X: IEEE 802.1X authentication is a Layer 2 authentication method.
- mab: MAC-Authentication Bypass is a Layer 2 authentication method.
- webauth: Web authentication is a Layer 3 authentication method.

Using this feature, you can control which ports use which authentication methods, and you can control the failover sequencing of methods on those ports. For example, MAC authentication bypass and 802.1x can be the primary or secondary authentication methods, and web authentication can be the fallback method if either or both of those authentication attempts fail.

The IEEE 802.1X Flexible Authentication feature supports the following host modes:

- multi-auth: Multiauthentication allows one authentication on a voice VLAN and multiple authentications on the data VLAN.
- multi-domain: Multidomain authentication allows two authentications: one on the voice VLAN and one on the data VLAN.

Open1x Authentication

Open1x authentication allows a device access to a port before that device is authenticated. When open authentication is configured, a new host can pass traffic according to the access control list (ACL) defined on the port. After the host is authenticated, the policies configured on the RADIUS server are applied to that host.

You can configure open authentication with these scenarios:

- Single-host mode with open authentication–Only one user is allowed network access before and after authentication.
- MDA mode with open authentication–Only one user in the voice domain and one user in the data domain are allowed.

- Multiple-hosts mode with open authentication–Any host can access the network.
- Multiple-authentication mode with open authentication–Similar to MDA, except multiple hosts can be authenticated.

Note If open authentication is configured, it takes precedence over other authentication controls. This means that if you use the **authentication open** interface configuration command, the port will grant access to the host irrespective of the **authentication port-control** interface configuration command.

Multidomain Authentication

The switch supports multidomain authentication (MDA), which allows both a data device and voice device, such as an IP phone, to authenticate on the same switch port. The port is divided into a data domain and a voice domain.

Note For all host modes, the line protocol stays up before authorization when port-based authentication is configured.

MDA does not enforce the order of device authentication. However, for best results, we recommend that a voice device is authenticated before a data device on an MDA-enabled port.

Follow these guidelines for configuring MDA:

- You must configure a switch port for MDA.
- You must configure the voice VLAN for the IP phone when the host mode is set to multidomain.
- Voice VLAN assignment on an MDA-enabled port is supported.
- To authorize a voice device, the AAA server must be configured to send a Cisco Attribute-Value (AV) pair attribute with a value of *device-traffic-class=voice*. Without this value, the switch treats the voice device as a data device.

Note When traffic-class=voice is downloaded from AAA servers as a service-template, a session will be created in DATA domain instead of VOICE domain.

- The guest VLAN and restricted VLAN features only apply to the data devices on an MDA-enabled port. The switch treats a voice device that fails authorization as a data device.
- If more than one device attempts authorization on either the voice or the data domain of a port, it is error disabled.
- Until a device is authorized, the port drops its traffic. Non-Cisco IP phones or voice devices are allowed into both the data and voice VLANs. The data VLAN allows the voice device to contact a DHCP server to obtain an IP address and acquire the voice VLAN information. After the voice device starts sending on the voice VLAN, its access to the data VLAN is blocked.
- A voice device MAC address that is binding on the data VLAN is not counted towards the port security MAC address limit.

- MDA can use MAC authentication bypass as a fallback mechanism to allow the switch port to connect to devices that do not support IEEE 802.1x authentication.
- When a *data* or a *voice* device is detected on a port, its MAC address is blocked until authorization succeeds. If the authorization fails, the MAC address remains blocked for 5 minutes.
- If more than five devices are detected on the *data* VLAN or more than one voice device is detected on the *voice* VLAN while a port is unauthorized, the port is error disabled.
- When a port host mode is changed from single- or multihost to multidomain mode, an authorized data device remains authorized on the port. However, a Cisco IP phone that has been allowed on the port voice VLAN is automatically removed and must be reauthenticated on that port.
- Active fallback mechanisms such as guest VLAN and restricted VLAN remain configured after a port changes from single- or multihost mode to multidomain mode.
- Switching a port host mode from multidomain to single- or multihost mode removes all authorized devices from the port.
- If a data domain is authorized first and placed in the guest VLAN, non-IEEE 802.1x-capable voice devices need to tag their packets on the voice VLAN to trigger authentication.
- We do not recommend per-user ACLs with an MDA-enabled port. An authorized device with a per-user ACL policy might impact traffic on both the voice and data VLANs of the port. If used, only one device on the port should enforce per-user ACLs.

802.1x Supplicant and Authenticator Switches with Network Edge Access Topology

The Network Edge Access Topology (NEAT) feature extends identity to areas outside the wiring closet (such as conference rooms). This allows any type of device to authenticate on the port.

- 802.1x switch supplicant: You can configure a switch to act as a supplicant to another switch by using the 802.1x supplicant feature. This configuration is helpful in a scenario, where, for example, a switch is outside a wiring closet and is connected to an upstream switch through a trunk port. A switch configured with the 802.1x switch supplicant feature authenticates with the upstream switch for secure connectivity. Once the supplicant switch authenticates successfully the port mode changes from access to trunk in an authenticator switch. In a supplicant switch you must manually configure trunk when enabling CISP.
- If the access VLAN is configured on the authenticator switch, it becomes the native VLAN for the trunk port after successful authentication.

In the default state, when you connect a supplicant switch to an authenticator switch that has BPDU guard enabled, the authenticator port could be error-disabled if it receives a Spanning Tree Protocol (STP) bridge protocol data unit (BPDU) packets before the supplicant switch has authenticated. You can control traffic exiting the supplicant port during the authentication period. Entering the **dot1x supplicant controlled transient** global configuration command temporarily blocks the supplicant port during authentication to ensure that the authenticator port does not shut down before authentication completes. If authentication fails, the supplicant port opens. Entering the **no dot1x supplicant controlled transient** global configuration command opens the supplicant port during the authentication period. This is the default behavior.

We strongly recommend using the **dot1x supplicant controlled transient**command on a supplicant switch when BPDU guard is enabled on the authenticator switch port with the **spanning-tree bpduguard enable** interface configuration command.

Note If you globally enable BPDU guard on the authenticator switch by using the **spanning-tree portfast bpduguard default** global configuration command, entering the **dot1x supplicant controlled transient** command does not prevent the BPDU violation.

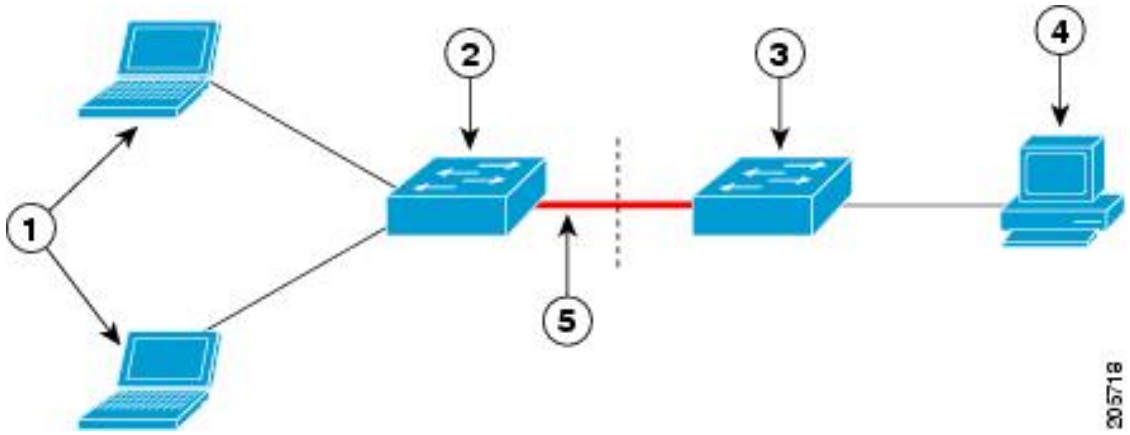
You can enable MDA or multiauth mode on the authenticator switch interface that connects to one more supplicant switches. Multihost mode is not supported on the authenticator switch interface.

When you reboot an authenticator switch with single-host mode enabled on the interface, the interface may move to err-disabled state before authentication. To recover from err-disabled state, flap the authenticator port to activate the interface again and initiate authentication.

Use the **dot1x supplicant force-multicast** global configuration command on the supplicant switch for Network Edge Access Topology (NEAT) to work in all host modes.

- Host Authorization: Ensures that only traffic from authorized hosts (connecting to the switch with supplicant) is allowed on the network. The switches use Client Information Signalling Protocol (CISP) to send the MAC addresses connecting to the supplicant switch to the authenticator switch.
- Auto enablement: Automatically enables trunk configuration on the authenticator switch, allowing user traffic from multiple VLANs coming from supplicant switches. Configure the cisco-av-pair as *device-traffic-class=switch* at the ISE. (You can configure this under the *group* or the *user* settings.)

Figure 112: Authenticator and Supplicant Switch Using CISP



1	Workstations (clients)	2	Supplicant switch (outside wiring closet)
3	Authenticator switch	4	Cisco ISE
5	Trunk port		

Note The **switchport nonegotiate** command is not supported on supplicant and authenticator switches with NEAT. This command should not be configured at the supplicant side of the topology. If configured on the authenticator side, the internal macros will automatically remove this command from the port.

802.1x User Distribution

You can configure 802.1x user distribution to load-balance users with the same group name across multiple different VLANs.

The VLANs are either supplied by the RADIUS server or configured through the switch CLI under a VLAN group name.

- Configure the RADIUS server to send more than one VLAN name for a user. The multiple VLAN names can be sent as part of the response to the user. The 802.1x user distribution tracks all the users in a particular VLAN and achieves load balancing by moving the authorized user to the least populated VLAN.
- Configure the RADIUS server to send a VLAN group name for a user. The VLAN group name can be sent as part of the response to the user. You can search for the selected VLAN group name among the VLAN group names that you configured by using the switch CLI. If the VLAN group name is found, the corresponding VLANs under this VLAN group name are searched to find the least populated VLAN. Load balancing is achieved by moving the corresponding authorized user to that VLAN.

  Whenever the RADIUS server sends a VLAN group name in the attribute as a result of authorization, the least populated VLAN out of the group is assigned to the end-user. In case of reauthentication (authentication session present) and CoA (session alive), the same VLAN is kept even if it is not the least populated VLAN in the group.

Note The RADIUS server can send the VLAN information in any combination of VLAN-IDs, VLAN names, or VLAN groups.

802.1x User Distribution Configuration Guidelines

- Confirm that at least one VLAN is mapped to the VLAN group.
- You can map more than one VLAN to a VLAN group.
- You can modify the VLAN group by adding or deleting a VLAN.
- When you clear an existing VLAN from the VLAN group name, none of the authenticated ports in the VLAN are cleared, but the mappings are removed from the existing VLAN group.
- If you clear the last VLAN from the VLAN group name, the VLAN group is cleared.
- You can clear a VLAN group even when the active VLANs are mapped to the group. When you clear a VLAN group, none of the ports or users that are in the authenticated state in any VLAN within the group are cleared, but the VLAN mappings to the VLAN group are cleared.

Network Admission Control Layer 2 IEEE 802.1x Validation

The switch supports the Network Admission Control (NAC) Layer 2 IEEE 802.1x validation, which checks the antivirus condition or *posture* of endpoint systems or clients before granting the devices network access. With NAC Layer 2 IEEE 802.1x validation, you can do these tasks:

- Download the Session-Timeout RADIUS attribute (Attribute[27]) and the Termination-Action RADIUS attribute (Attribute[29]) from the authentication server.

- Set the number of seconds between reauthentication attempts as the value of the Session-Timeout RADIUS attribute (Attribute[27]) and get an access policy against the client from the RADIUS server.
- Set the action to be taken when the switch tries to reauthenticate the client by using the Termination-Action RADIUS attribute (Attribute[29]). If the value is the *DEFAULT* or is not set, the session ends. If the value is RADIUS-Request, the reauthentication process starts.
- Set the list of VLAN number or name or VLAN group name as the value of the Tunnel Group Private ID (Attribute[81]) and the preference for the VLAN number or name or VLAN group name as the value of the Tunnel Preference (Attribute[83]). If you do not configure the Tunnel Preference, the first Tunnel Group Private ID (Attribute[81]) attribute is picked up from the list.
- View the NAC posture token, which shows the posture of the client, by using the **show authentication** privileged EXEC command.
- Configure secondary private VLANs as guest VLANs.

Configuring NAC Layer 2 IEEE 802.1x validation is similar to configuring IEEE 802.1x port-based authentication except that you must configure a posture token on the RADIUS server.

Voice Aware 802.1x Security

Note To use voice aware IEEE 802.1x authentication, the switch must be running the LAN base image.

You use the voice aware 802.1x security feature to configure the switch to disable only the VLAN on which a security violation occurs, whether it is a data or voice VLAN. In previous releases, when an attempt to authenticate the data client caused a security violation, the entire port shut down, resulting in a complete loss of connectivity.

You can use this feature in IP phone deployments where a PC is connected to the IP phone. A security violation found on the data VLAN results in the shutdown of only the data VLAN. The traffic on the voice VLAN flows through the switch without interruption.

Common Session ID

Authentication manager uses a single session ID (referred to as a common session ID) for a client no matter which authentication method is used. This ID is used for all reporting purposes, such as the show commands and MIBs. The session ID appears with all per-session syslog messages.

The session ID includes:

- The IP address of the Network Access Device (NAD)
- A monotonically increasing unique 32 bit integer
- The session start time stamp (a 32 bit integer)

This example shows how the session ID appears in the output of the **show authentication** command. The session ID in this example is 160000050000000B288508E5:

```
Device# show authentication
Interface  MAC Address     Method   Domain   Status         Session ID
Fa4/0/4    0000.0000.0203  mab      DATA     Authz Success  160000050000000B288508E5
```

This is an example of how the session ID appears in the syslog output. The session ID in this example is also160000050000000B288508E5:

```
1w0d: %AUTHMGR-5-START: Starting 'mab' for client (0000.0000.0203) on Interface Fa4/0/4
AuditSessionID 160000050000000B288508E5
1w0d: %MAB-5-SUCCESS: Authentication successful for client (0000.0000.0203) on Interface
Fa4/0/4 AuditSessionID 160000050000000B288508E5
1w0d: %AUTHMGR-7-RESULT: Authentication result 'success' from 'mab' for client
(0000.0000.0203) on Interface Fa4/0/4 AuditSessionID 160000050000000B288508E5
```

The session ID is used by the NAD, the AAA server, and other report-analyzing applications to identify the client. The ID appears automatically. No configuration is required.

Maximum Number of Allowed Devices Per Port

This is the maximum number of devices allowed on an 802.1x-enabled port:

- In single-host mode, only one device is allowed on the access VLAN. If the port is also configured with a voice VLAN, an unlimited number of Cisco IP phones can send and receive traffic through the voice VLAN.
- In multidomain authentication (MDA) mode, one device is allowed for the access VLAN, and one IP phone is allowed for the voice VLAN.
- In multihost mode, only one 802.1x supplicant is allowed on the port, but an unlimited number of non-802.1x hosts are allowed on the access VLAN. An unlimited number of devices are allowed on the voice VLAN.

How to Configure IEEE 802.1x Port-Based Authentication

Configuring 802.1x Authentication

To allow per-user ACLs or VLAN assignment, you must enable AAA authorization to configure the switch for all network-related service requests.

This is the 802.1x AAA process:

Before you begin

To configure 802.1x port-based authentication, you must enable authentication, authorization, and accounting (AAA) and specify the authentication method list. A method list describes the sequence and authentication method to be queried to authenticate a user.

Procedure

Step 1 A user connects to a port on the switch.

Step 2 Authentication is performed.

Step 3 VLAN assignment is enabled, as appropriate, based on the RADIUS server configuration.

Step 4 The switch sends a start message to an accounting server.

Step 5 Re-authentication is performed, as necessary.

Step 6 The switch sends an interim accounting update to the accounting server that is based on the result of reauthentication.

Step 7 The user disconnects from the port.

Step 8 The switch sends a stop message to the accounting server.

Configuring 802.1x Port-Based Authentication

Perform these steps to configure 802.1x port-based authentication:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**aaa authentication dot1x** {**default**} *method1* **Example:** `Device(config)# aaa authentication dot1x default group radius`	Creates an 802.1x authentication method list. To create a default list that is used when a named list is *not* specified in the **authentication** command, use the **default** keyword followed by the method that is to be used in default situations. The default method list is automatically applied to all ports. For *method1*, enter the **group radius** keywords to use the list of all RADIUS servers for authentication. **Note** Though other keywords are visible in the command-line help string, only the **group radius** keywords are supported.

	Command or Action	Purpose
Step 5	**dot1x system-auth-control** **Example:** `Device(config)# dot1x system-auth-control`	Enables 802.1x authentication globally on the switch.
Step 6	**aaa authorization network** {**default**} **group radius** **Example:** `Device(config)# aaa authorization network default group radius`	(Optional) Configures the switch to use user-RADIUS authorization for all network-related service requests, such as per-user ACLs or VLAN assignment.
Step 7	**radius server** *server name* **Example:** `Device(config)# radius server rsim address ipv4 124.2.2.12`	(Optional) Specifies the IP address of the RADIUS server.
Step 8	**address** {**ipv4 | ipv6**} *ip address* **Example:** `Device(config-radius-server)# address ipv4 10.0.1.12`	Configures the IP address for the RADIUS server.
Step 9	**key** *string* **Example:** `Device(config-radius-server)# key rad123`	(Optional) Specifies the authentication and encryption key used between the switch and the RADIUS daemon running on the RADIUS server.
Step 10	**exit** **Example:** `Device(config-radius-server)# exit`	Exits the RADIUS server mode and enters the global configuration mode.
Step 11	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port connected to the client that is to be enabled for IEEE 802.1x authentication, and enter interface configuration mode.

	Command or Action	Purpose
Step 12	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	(Optional) Sets the port to access mode only if you configured the RADIUS server in Step 6 and Step 7.
Step 13	**authentication port-control auto** **Example:** `Device(config-if)# authentication port-control auto`	Enables 802.1x authentication on the port.
Step 14	**dot1x pae authenticator** **Example:** `Device(config-if)# dot1x pae authenticator`	Sets the interface Port Access Entity to act only as an authenticator and ignore messages meant for a supplicant.
Step 15	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Periodic Reauthentication

You can enable periodic 802.1x client reauthentication and specify how often it occurs. If you do not specify a time period before enabling reauthentication, the number of seconds between attempts is 3600.

Follow these steps to enable periodic reauthentication of the client and to configure the number of seconds between reauthentication attempts. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enter interface configuration mode.
Step 4	**authentication periodic** **Example:** `Device(config-if)# authentication periodic`	Enables periodic reauthentication of the client, which is disabled by default. **Note** The default value is 3600 seconds. To change the value of the reauthentication timer or to have the switch use a RADIUS-provided session timeout, enter the **authentication timer reauthenticate** command.
Step 5	**authentication timer** {{[**inactivity** | **reauthenticate** | **restart** | **unauthorized**]} {*value*}} **Example:** `Device(config-if)# authentication timer reauthenticate 180`	Sets the number of seconds between reauthentication attempts. The **authentication timer** keywords have these meanings: • **inactivity**: Interval in seconds after which if there is no activity from the client then it is unauthorized • **reauthenticate**: Time in seconds after which an automatic reauthentication attempt is initiated • **restart** *value*: Interval in seconds after which an attempt is made to authenticate an unauthorized port • **unauthorized** *value*: Interval in seconds after which an unauthorized session will get deleted This command affects the behavior of the switch only if periodic reauthentication is enabled.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring 802.1x Violation Modes

You can configure an 802.1x port so that it shuts down, generates a syslog error, or discards packets from a new device when:

- A device connects to an 802.1x-enabled port
- The maximum number of allowed about devices have been authenticated on the port

Beginning in privileged EXEC mode, follow these steps to configure the security violation actions on the switch:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**aaa authentication dot1x** {**default**} *method1* **Example:** `Device(config)# aaa authentication dot1x default group radius`	Creates an 802.1x authentication method list. To create a default list that is used when a named list is *not* specified in the **authentication** command, use the **default** keyword followed by the method that is to be used in default situations. The default method list is automatically applied to all ports. For *method1*, enter the **group radius** keywords to use the list of all RADIUS servers for authentication.
Step 5	**interface** *interface-type interface-number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port connected to the client that is to be enabled for IEEE 802.1x authentication, and enter interface configuration mode.

	Command or Action	Purpose
Step 6	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Sets the port to access mode.
Step 7	**authentication violation** {**shutdown** | **restrict** | **protect** | **replace**} **Example:** `Device(config-if)# authentication violation restrict`	Configures the violation mode. The keywords have these meanings: • **shutdown**: Error disable the port. • **restrict**: Generate a syslog error. • **protect**: Drop packets from any new device that sends traffic to the port. • **replace**: Removes the current session and authenticates with the new host.
Step 8	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Changing the Quiet Period

When the switch cannot authenticate the client, the switch remains idle for a set period of time and then tries again. The **authentication timer restart** interface configuration command controls the idle period. A failed authentication of the client might occur because the client provided an invalid password. You can provide a faster response time to the user by entering a number smaller than the default.

Beginning in privileged EXEC mode, follow these steps to change the quiet period. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**authentication timer restart** *seconds* **Example:** `Device(config-if)# authentication timer restart 30`	Sets the number of seconds that the switch remains in the quiet state following a failed authentication exchange with the client. The range is 1 to 1073741823 seconds; the default is 60.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 6	**show authentication sessions interface** *interface-id* **Example:** `Device# show authentication sessions interface gigabitethernet1/1`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Changing the Switch-to-Client Retransmission Time

The client responds to the EAP-request/identity frame from the switch with an EAP-response/identity frame. If the switch does not receive this response, it waits a set period of time (known as the retransmission time) and then resends the frame.

Note You should change the default value of this command only to adjust for unusual circumstances such as unreliable links or specific behavioral problems with certain clients and authentication servers.

Beginning in privileged EXEC mode, follow these steps to change the amount of time that the switch waits for client notification. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**authentication timer reauthenticate** *seconds* **Example:** `Device(config-if)# authentication timer reauthenticate 60`	Sets the number of seconds that the switch waits for a response to an EAP-request/identity frame from the client before resending the request. The range is 1 to 1073741823 seconds; the default is 5.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 6	**show authentication sessions interface** *interface-id* **Example:** `Device# show authentication sessions interface gigabitethernet 1/1`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config`	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`startup-config`	

Setting the Switch-to-Client Frame-Retransmission Number

In addition to changing the switch-to-client retransmission time, you can change the number of times that the switch sends an EAP-request/identity frame (assuming no response is received) to the client before restarting the authentication process.

Note You should change the default value of this command only to adjust for unusual circumstances such as unreliable links or specific behavioral problems with certain clients and authentication servers.

Beginning in privileged EXEC mode, follow these steps to set the switch-to-client frame-retransmission number. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device># enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**dot1x max-reauth-req** *count* **Example:** `Device(config-if)# dot1x max-reauth-req 5`	Sets the number of times that the switch sends an EAP-request/identity frame to the client before restarting the authentication process. The range is 1 to 10; the default is 2.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config-if)#end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Host Mode

Beginning in privileged EXEC mode, follow these steps to allow multiple hosts (clients) on an IEEE 802.1x-authorized port that has the **authentication port-control** interface configuration command set to **auto**. Use the **multi-domain** keyword to configure and enable multidomain authentication (MDA), which allows both a host and a voice device, such as an IP phone (Cisco or non-Cisco), on the same switch port. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to which multiple hosts are indirectly attached, and enters interface configuration mode.
Step 4	**authentication host-mode** [**multi-auth** | **multi-domain** | **multi-host** | **single-host**] **Example:** `Device(config-if)# authentication host-mode multi-host`	Allows multiple hosts (clients) on an 802.1x-authorized port. The keywords have these meanings: • **multi-auth**: Allow multiple authenticated clients on both the voice VLAN and data VLAN. **Note**

	Command or Action	Purpose
		The **multi-auth** keyword is only available with the **authentication host-mode** command. • **multi-host**: Allow multiple hosts on an 802.1x-authorized port after a single host has been authenticated. • **multi-domain**: Allow both a host and a voice device, such as an IP phone (Cisco or non-Cisco), to be authenticated on an IEEE 802.1x-authorized port. **Note** You must configure the voice VLAN for the IP phone when the host mode is set to **multi-domain**. Make sure that the **authentication port-control** interface configuration command is set to **auto** for the specified interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Enabling MAC Move

MAC move allows an authenticated host to move from one port on the device to another.

Beginning in privileged EXEC mode, follow these steps to globally enable MAC move on the device. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**authentication mac-move permit** **Example:** `Device(config)# authentication mac-move permit`	Enables MAC move on the device. Default is deny. In Session Aware Networking mode, the default CLI is **access-session mac-move deny**. To enable Mac Move in Session Aware Networking, use the **no access-session mac-move** global configuration command. In legacy mode (IBNS 1.0), default value for **mac-move** is **deny** and in C3PL mode (IBNS 2.0) default value is **permit**.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Enabling MAC Replace

MAC replace allows a host to replace an authenticated host on a port.

Beginning in privileged EXEC mode, follow these steps to enable MAC replace on an interface. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Devic(config)# interface`	Specifies the port to be configured, and enters interface configuration mode.

	Command or Action	Purpose
	gigabitethernet1/1	
Step 4	**authentication violation** {**protect** | **replace** | **restrict** | **shutdown**} **Example:** `Device(config-if)# authentication violation replace`	Use the **replace** keyword to enable MAC replace on the interface. The port removes the current session and initiates authentication with the new host. The other keywords have these effects: • **protect**: the port drops packets with unexpected MAC addresses without generating a system message. • **restrict**: violating packets are dropped by the CPU and a system message is generated. • **shutdown**: the port is error disabled when it receives an unexpected MAC address.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring 802.1x Accounting

Enabling AAA system accounting with 802.1x accounting allows system reload events to be sent to the accounting RADIUS server for logging. The server can then infer that all active 802.1x sessions are closed.

Because RADIUS uses the unreliable UDP transport protocol, accounting messages might be lost due to poor network conditions. If the switch does not receive the accounting response message from the RADIUS server after a configurable number of retransmissions of an accounting request, this system message appears:

```
Accounting message %s for session %s failed to receive Accounting Response.
```

When the stop message is not sent successfully, this message appears:

```
00:09:55: %RADIUS-4-RADIUS_DEAD: RADIUS server 172.20.246.201:1645,1646 is not responding.
```

Note You must configure the RADIUS server to perform accounting tasks, such as logging start, stop, and interim-update messages and time stamps. To turn on these functions, enable logging of "Update/Watchdog packets from this AAA client" in your RADIUS server Network Configuration tab. Next, enable "CVS RADIUS Accounting" in your RADIUS server System Configuration tab.

Beginning in privileged EXEC mode, follow these steps to configure 802.1x accounting after AAA is enabled on your switch. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**aaa accounting dot1x default start-stop group radius** **Example:** `Device(config-if)# aaa accounting dot1x default start-stop group radius`	Enables 802.1x accounting using the list of all RADIUS servers.
Step 5	**aaa accounting system default start-stop group radius** **Example:** `Device(config-if)# aaa accounting system default start-stop group radius`	(Optional) Enables system accounting (using the list of all RADIUS servers) and generates system accounting reload event messages when the switch reloads.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring 802.1x Readiness Check

The 802.1x readiness check monitors 802.1x activity on all the switch ports and displays information about the devices connected to the ports that support 802.1x. You can use this feature to determine if the devices connected to the switch ports are 802.1x-capable.

The 802.1x readiness check is allowed on all ports that can be configured for 802.1x. The readiness check is not available on a port that is configured as **dot1x force-unauthorized**.

Follow these steps to enable the 802.1x readiness check on the switch:

Before you begin

Follow these guidelines to enable the readiness check on the switch:

- The readiness check is typically used before 802.1x is enabled on the switch.
- When you configure the **dot1x test eapol-capable** command on an 802.1x-enabled port, and the link comes up, the port queries the connected client about its 802.1x capability. When the client responds with a notification packet, it is 802.1x-capable. A syslog message is generated if the client responds within the timeout period. If the client does not respond to the query, the client is not 802.1x-capable. No syslog message is generated.
- The readiness check can be sent on a port that handles multiple hosts (for example, a PC that is connected to an IP phone). A syslog message is generated for each of the clients that respond to the readiness check within the timer period.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**dot1x test eapol-capable** [**interface** *interface-id*] **Example:** `Device# dot1x test eapol-capable interface gigabitethernet1/1` `DOT1X_PORT_EAPOL_CAPABLE:DOT1X: MAC 00-01-02-4b-f1-a3 on gigabitethernet1/1 is EAPOL capable`	Enables the 802.1x readiness check on the switch. (Optional) For *interface-id* specify the port on which to check for IEEE 802.1x readiness. **Note** If you omit the optional **interface** keyword, all interfaces on the switch are tested.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 4	**dot1x test timeout** *timeout* **Example:** `Device(config)# dot1x test timeout 54`	(Optional) Configures the timeout used to wait for EAPOL response. The range is from 1 to 65535 seconds. The default is 10 seconds.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Switch-to-RADIUS Server Communication

Follow these steps to configure the RADIUS server parameters:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip radius source-interface vlan** *vlan interface number* **Example:** `Device(config)# ip radius source-interface vlan 80`	Specifies that the RADIUS packets have the IP address of the indicated interface.
Step 4	**radius server** *server name* **Example:** `Device(config)# radius server rsim address ipv4 172.16.0.1`	(Optional) Specifies the IP address of the RADIUS server.

	Command or Action	Purpose
Step 5	**address** {**ipv4** | **ipv6**} *ip address* **Example:** `Device(config-radius-server)# address ipv4 10.0.1.2 auth-port 1550 acct-port 1560`	Configures the IP address for the RADIUS server.
Step 6	**key** *string* **Example:** `Device(config-radius-server)# key rad123`	(Optional) Specifies the authentication and encryption key used between the switch and the RADIUS daemon running on the RADIUS server.
Step 7	**exit** **Example:** `Device(config-radius-server)# exit`	Exits the RADIUS server mode and enters the global configuration mode.
Step 8	**radius-server dead-criteria tries** *num-tries* **Example:** `Device(config)# radius-server dead-criteria tries 30`	Specifies the number of unanswered sent messages to a RADIUS server before considering the server to be inactive. The range of *num-tries* is 1 to 100.
Step 9	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Setting the Reauthentication Number

You can also change the number of times that the device restarts the authentication process before the port changes to the unauthorized state.

Note You should change the default value of this command only to adjust for unusual circumstances such as unreliable links or specific behavioral problems with certain clients and authentication servers.

Beginning in privileged EXEC mode, follow these steps to set the reauthentication number. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device# interface gigabitethernet1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Sets the port to access mode only if you previously configured the RADIUS server.
Step 5	**dot1x max-req** *count* **Example:** `Device(config-if)# dot1x max-req 4`	Sets the number of times that the device restarts the authentication process before the port changes to the unauthorized state. The range is 0 to 10; the default is 2.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring a Guest VLAN

When you configure a guest VLAN, clients that are not 802.1x-capable are put into the guest VLAN when the server does not receive a response to its EAP request/identity frame. Clients that are 802.1x-capable but that fail authentication are not granted network access. The switch supports guest VLANs in single-host or multiple-hosts mode.

Beginning in privileged EXEC mode, follow these steps to configure a guest VLAN. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**dot1x port-control auto** **Example:** `Device(config-if)# dot1x port-control auto`	Enables 802.1x authentication on the port.
Step 5	**authentication event no-response action authorize vlan** *vlan-id* **Example:** `Device(config-if)# authentication event no-response action authorize vlan 2`	Specifies an active VLAN as an 802.1x guest VLAN. The range is 1 to 4094. You can configure any active VLAN except an internal VLAN (routed port), an RSPAN VLAN or a voice VLAN as an 802.1x guest VLAN.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring a Restricted VLAN

When you configure a restricted VLAN on a device, clients that are IEEE 802.1x-compliant are moved into the restricted VLAN when the authentication server does not receive a valid username and password. The device supports restricted VLANs only in single-host mode.

Beginning in privileged EXEC mode, follow these steps to configure a restricted VLAN. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**authentication port-control auto** **Example:** `Device(config-if)# authentication port-control auto`	Enables 802.1x authentication on the port.
Step 5	**authentication event fail action authorize vlan** *vlan-id* **Example:** `Device(config-if)# authentication event fail action authorize vlan 2`	Specifies an active VLAN as an 802.1x restricted VLAN. The range is 1 to 4094. You can configure any active VLAN except an internal VLAN (routed port), an RSPAN VLAN or a voice VLAN as an 802.1x restricted VLAN.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring the Number of Authentication Attempts on a Restricted VLAN

You can configure the maximum number of authentication attempts allowed before a user is assigned to the restricted VLAN by using the **authentication event fail retry** *retry count* interface configuration command. The range of allowable authentication attempts is 1 to 3. The default is 3 attempts.

Beginning in privileged EXEC mode, follow these steps to configure the maximum number of allowed authentication attempts. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**authentication port-control auto** **Example:** `Device(config-if)# authentication port-control auto`	Enables 802.1x authentication on the port.
Step 5	**authentication event fail action authorize vlan** *vlan-id* **Example:** `Device(config-if)# authentication event fail action authorize vlan 8`	Specifies an active VLAN as an 802.1x restricted VLAN. The range is 1 to 4094. You can configure any active VLAN except an internal VLAN (routed port), an RSPAN VLAN or a voice VLAN as an 802.1x restricted VLAN.
Step 6	**authentication event fail retry** *retry count* **Example:** `Device(config-if)# authentication event`	Specifies a number of authentication attempts before a port moves to the auth fail VLAN.

	Command or Action	Purpose
	`fail retry 2`	
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring 802.1x Inaccessible Authentication Bypass with Critical Voice VLAN

Beginning in privileged EXEC mode, follow these steps to configure critical voice VLAN on a port and enable the inaccessible authentication bypass feature.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**radius-server dead-criteria** {**time** *seconds* } [**tries** *number*] **Example:** `Device(config)# radius-server dead-criteria time 20 tries 10`	Sets the conditions that determine when a RADIUS server is considered un-available or down (dead). • **time**: 1 to 120 seconds. The switch dynamically determines a default *seconds* value between 10 and 60. • **number**: 1 to 100 tries. The switch dynamically determines a default **tries***number* between 10 and 100.

	Command or Action	Purpose
Step 5	**radius-serverdeadtime***minutes* **Example:** `Device(config)# radius-server deadtime 60`	(Optional) Sets the number of minutes during which a RADIUS server is not sent requests. The range is from 0 to 1440 minutes (24 hours). The default is 0 minutes.
Step 6	**radius server** *server name* **Example:** `Device(config)# radius server rsim address ipv4 124.2.2.12`	(Optional) Specifies the IP address of the RADIUS server.
Step 7	**address** {**ipv4** | **ipv6**} *ip address* **auth-port** *port_number* **acct-port** *port_number* **Example:** `Device(config-radius-server)# address ipv4 10.0.1.2 auth-port 1550 acct-port 1560`	Configures the IP address for the RADIUS server.
Step 8	**key** *string* **Example:** `Device(config-radius-server)# key rad123`	(Optional) Specifies the authentication and encryption key used between the switch and the RADIUS daemon running on the RADIUS server.
Step 9	**exit** **Example:** `Device(config-radius-server)# exit`	Exits the RADIUS server mode and enters the global configuration mode.
Step 10	**dot1x critical** {**eapol** | **recovery delay** *milliseconds*} **Example:** `Device(config)# dot1x critical eapol` `Device(config)# dot1x critical recovery delay 2000`	(Optional) Configure the parameters for inaccessible authentication bypass: • **eapol**: Specify that the switch sends an EAPOL-Success message when the switch successfully authenticates the critical port. • **recovery delay***milliseconds*: Set the recovery delay period during which the switch waits to re-initialize a critical port when a RADIUS server that was unavailable becomes available. The range is from 1 to 10000 milliseconds. The

	Command or Action	Purpose
		default is 1000 milliseconds (a port can be re-initialized every second).
Step 11	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specify the port to be configured, and enters interface configuration mode.
Step 12	**authentication event server dead action** {**authorize** | **reinitialize**} **vlan** *vlan-id*] **Example:** `Device(config-if)# authentication event server dead action reinitialicze vlan 20`	Use these keywords to move hosts on the port if the RADIUS server is unreachable: • **authorize**: Move any new hosts trying to authenticate to the user-specified critical VLAN. • **reinitialize**: Move all authorized hosts on the port to the user-specified critical VLAN.
Step 13	**switchport voice vlan** *vlan-id* **Example:** `Device(config-if)# switchport voice vlan`	Specifies the voice VLAN for the port. The voice VLAN cannot be the same as the critical data VLAN configured in Step 6.
Step 14	**authentication event server dead action authorize voice** **Example:** `Device(config-if)# authentication event server dead action authorize voice`	Configures critical voice VLAN to move data traffic on the port to the voice VLAN if the RADIUS server is unreachable.
Step 15	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 16	**show authentication sessions interface** *interface-id* **Example:** `Device(config-if)# show authentication sessions interface gigabitethernet 1/1`	(Optional) Verify your entries.

Example

To return to the RADIUS server default settings, use the **no radius-server dead-criteria**, the **no radius-server deadtime**, and the **no radius server** global configuration commands. To disable inaccessible authentication bypass, use the **no authentication event server dead action** interface configuration command. To disable critical voice VLAN, use the no **authentication event server dead action authorize voice** interface configuration command.

Configuring MAC Authentication Bypass

Beginning in privileged EXEC mode, follow these steps to enable MAC authentication bypass. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**authentication port-control auto** **Example:** `Device(config-if)# authentication port-control auto`	Enables 802.1x authentication on the port.
Step 5	**mab** [**eap**] **Example:** `Device(config-if)# mab`	Enables MAC authentication bypass. (Optional) Use the **eap** keyword to configure the device to use EAP for authorization.

	Command or Action	Purpose
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring 802.1x User Distribution

Beginning in privileged EXEC mode, follow these steps to configure a VLAN group and to map a VLAN to it:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan group** *vlan-group-name* **vlan-list** *vlan-list* **Example:** `Device(config)# vlan group eng-dept vlan-list 10`	Configures a VLAN group, and maps a single VLAN or a range of VLANs to it.
Step 4	**no vlan group** *vlan-group-name* **vlan-list** *vlan-list* **Example:** `Device(config)# no vlan group eng-dept vlan-list 10`	Clears the VLAN group configuration or elements of the VLAN group configuration.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring NAC Layer 2 802.1x Validation

You can configure NAC Layer 2 802.1x validation, which is also referred to as 802.1x authentication with a RADIUS server.

Beginning in privileged EXEC mode, follow these steps to configure NAC Layer 2 802.1x validation. The procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Sets the port to access mode only if you configured the RADIUS server.
Step 5	**authentication event no-response action authorize vlan** *vlan-id* **Example:** `Device(config-if)# authentication event no-response action authorize vlan 8`	Specifies an active VLAN as an 802.1x guest VLAN. The range is 1 to 4094. You can configure any active VLAN except an internal VLAN (routed port), an RSPAN VLAN, or a voice VLAN as an 802.1x guest VLAN.
Step 6	**authentication periodic** **Example:** `Device(config-if)# authentication periodic`	Enables periodic reauthentication of the client, which is disabled by default.

	Command or Action	Purpose
Step 7	**authentication timer reauthenticate** **Example:** `Device(config-if)# authentication timer reauthenticate`	Sets reauthentication attempt for the client (set to one hour). This command affects the behavior of the switch only if periodic reauthentication is enabled.
Step 8	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 9	**show authentication sessions interface** *interface-id* **Example:** `Device# show authentication sessions interface gigabitethernet1/1`	Displays information about current Auth Manager sessions on the interface.

Configuring an Authenticator Switch with NEAT

Configuring this feature requires that one switch outside a wiring closet is configured as a supplicant and is connected to an authenticator switch.

Note

- The authenticator switch interface configuration must be restored to access mode by explicitly flapping it if a line card is removed and inserted in the chassis when CISP or NEAT session is active.
- The *cisco-av-pairs* must be configured as *device-traffic-class=switch* on the ISE, which sets the interface as a trunk after the supplicant is successfully authenticated.

Beginning in privileged EXEC mode, follow these steps to configure a switch as an authenticator:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**cisp enable** **Example:** `Device(config)# cisp enable`	Enables CISP.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 5	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Sets the port mode to **access**.
Step 6	**authentication port-control auto** **Example:** `Device(config-if)# authentication port-control auto`	Sets the port-authentication mode to auto.
Step 7	**dot1x pae authenticator** **Example:** `Device(config-if)# dot1x pae authenticator`	Configures the interface as a port access entity (PAE) authenticator.
Step 8	**spanning-tree portfast** **Example:** `Device(config-if)# spanning-tree portfast trunk`	Enables Port Fast on an access port connected to a single workstation or server..
Step 9	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring a Supplicant Switch with NEAT

Beginning in privileged EXEC mode, follow these steps to configure a switch as a supplicant:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cisp enable** **Example:** `Device(config)# cisp enable`	Enables CISP.
Step 4	**dot1x credential**s *profile* **Example:** `Device(config)# dot1x credentials test`	Creates 802.1x credentials profile. This must be attached to the port that is configured as supplicant.
Step 5	**username** *suppswitch* **Example:** `Device(config)# username suppswitch`	Creates a username.
Step 6	**password** *password* **Example:** `Device(config)# password myswitch`	Creates a password for the new username.
Step 7	**dot1x supplicant force-multicast** **Example:** `Device(config)# dot1x supplicant force-multicast`	Forces the switch to send only multicast EAPOL packets when it receives either unicast or multicast packets. This also allows NEAT to work on the supplicant switch in all host modes.

	Command or Action	Purpose
Step 8	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 9	**switchport trunk encapsulation dot1q** **Example:** `Device(config-if)# switchport trunk encapsulation dot1q`	Sets the port to trunk mode.
Step 10	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Configures the interface as a VLAN trunk port.
Step 11	**dot1x pae supplicant** **Example:** `Device(config-if)# dot1x pae supplicant`	Configures the interface as a port access entity (PAE) supplicant.
Step 12	**dot1x credentials** *profile-name* **Example:** `Device(config-if)# dot1x credentials test`	Attaches the 802.1x credentials profile to the interface.
Step 13	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring 802.1x Authentication with Downloadable ACLs and Redirect URLs

Note You must configure a downloadable ACL on the ACS before downloading it to the switch.

After authentication on the port, you can use the **show ip access-list** privileged EXEC command to display the downloaded ACLs on the port.

Note The output of the **show ip access-lists interface** command does not display dACL or ACL filter IDs. This is because the ACLs are attached to the virtual ports created by multidomain authentication for each authentication session; instead of the physical interface. To display dACL or ACL filter IDs, use the **show ip access-lists** *access-list-name* command. The *access-list-name* should be taken from the **show authentication sessions interface** *interface-name* **detail** command output. The *access-list-name* is case sensitive.

Configuring Downloadable ACLs

The policies take effect after client authentication and the client IP address addition to the IP device tracking table. The switch then applies the downloadable ACL to the port.

Beginning in privileged EXEC mode:

Before you begin

SISF-Based device tracking is a prerequisite to configuring 802.1x authentication. Ensure that you have enabled device tracking programmatically or manually. For more information, see the *Confguring SISF-Based Tracking* chapter.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**aaa authorization network default local group radius** **Example:** `Device(config)# aaa authorization network default local group radius`	Sets the authorization method to local. To remove the authorization method, use the **no aaa authorization network default local group radius** command.

	Command or Action	Purpose
Step 5	**radius-server vsa send authentication** **Example:** `Device(config)# radius-server vsa send authentication`	Configures the radius vsa send authentication.
Step 6	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 7	**ip access-group** *acl-id* **in** **Example:** `Device(config-if)# ip access-group default_acl in`	Configures the default ACL on the port in the input direction. **Note** The *acl-id* is an access list name or number.
Step 8	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring a Downloadable Policy

Before you begin

SISF-Based device tracking is a prerequisite to configuring 802.1x authentication. Ensure that you have enabled device tracking programmatically or manually.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**access-list** *access-list-number* **{ deny | permit } { hostname | any | host } log** **Example:** `Device(config)# access-list 1 deny any log`	Defines the default port ACL. The access-list-number is a decimal number from 1 to 99 or 1300 to 1999. Enter **deny** or **permit** to specify whether to deny or permit access if conditions are matched. The source is the source address of the network or host that sends a packet, such as this: • **hostname**: The 32-bit quantity in dotted-decimal format. • **any**: The keyword any as an abbreviation for source and source-wildcard value of 0.0.0.0 255.255.255.255. You do not need to enter a source-wildcard value. • **host**: The keyword host as an abbreviation for source and source-wildcard of source 0.0.0.0. (Optional) Applies the source-wildcard wildcard bits to the source. (Optional) Enters log to cause an informational logging message about the packet that matches the entry to be sent to the console.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode.
Step 5	**ip access-group** *acl-id* **in** **Example:** `Device(config-if)# ip access-group default_acl in`	Configures the default ACL on the port in the input direction. **Note** The *acl-id* is an access list name or number.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 7	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 8	**aaa authorization network default group radius** **Example:** `Device(config)# aaa authorization network default group radius`	Sets the authorization method to local. To remove the authorization method, use the **no aaa authorization network default group radius** command.
Step 9	**radius-server vsa send authentication** **Example:** `Device(config)# radius-server vsa send authentication`	Configures the network access server to recognize and use vendor-specific attributes. **Note** The downloadable ACL must be operational.
Step 10	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring VLAN ID Based MAC Authentication

Beginning in privileged EXEC mode, follow these steps:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**mab request format attribute 32 vlan access-vlan** **Example:** `Device(config)# mab request format attribute 32 vlan access-vlan`	Enables VLAN ID-based MAC authentication.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Flexible Authentication Ordering

The examples used in the instructions below changes the order of Flexible Authentication Ordering so that MAB is attempted before IEEE 802.1X authentication (dot1x). MAB is configured as the first authentication method, so MAB will have priority over all other authentication methods.

Beginning in privileged EXEC mode, follow these steps:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**switchport mode access** **Example:**	Sets the port to access mode only if you previously configured the RADIUS server.

	Command or Action	Purpose
	`Device(config-if)# switchport mode access`	
Step 5	**authentication order** [**dot1x** | **mab**] | {**webauth**} **Example:** `Device(config-if)# authentication order mab dot1x`	(Optional) Sets the order of authentication methods used on a port.
Step 6	**authentication priority** [**dot1x** | **mab**] | {**webauth**} **Example:** `Device(config-if)# authentication priority mab dot1x`	(Optional) Adds an authentication method to the port-priority list.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Open1x

Beginning in privileged EXEC mode, follow these steps to enable manual control of the port authorization state:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 4	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Sets the port to access mode only if you configured the RADIUS server.
Step 5	**authentication control-direction** {**both** | **in**} **Example:** `Device(config-if)# authentication control-direction both`	(Optional) Configures the port control as unidirectional or bidirectional.
Step 6	**authentication fallback** *name* **Example:** `Device(config-if)# authentication fallback profile1`	(Optional) Configures a port to use web authentication as a fallback method for clients that do not support 802.1x authentication.
Step 7	**authentication host-mode** [**multi-auth** | **multi-domain** | **multi-host** | **single-host**] **Example:** `Device(config-if)# authentication host-mode multi-auth`	(Optional) Sets the authorization manager mode on a port.
Step 8	**no authentication closed** **Example:** `Device(config-if)# no authentication closed`	(Optional) Enables or disable open access on a port.
Step 9	**authentication order** [**dot1x** | **mab**] | {**webauth**} **Example:** `Device(config-if)# authentication order dot1x webauth`	(Optional) Sets the order of authentication methods used on a port.

	Command or Action	Purpose
Step 10	**authentication periodic** **Example:** `Device(config-if)# authentication periodic`	(Optional) Enables or disable reauthentication on a port.
Step 11	**authentication port-control** {**auto** | **force-authorized** | **force-un authorized**} **Example:** `Device(config-if)# authentication port-control auto`	(Optional) Enables manual control of the port authorization state.
Step 12	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Disabling 802.1x Authentication on a Port

You can disable 802.1x authentication on the port by using the **no dot1x pae** interface configuration command.

Beginning in privileged EXEC mode, follow these steps to disable 802.1x authentication on the port. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet`	Specifies the port to be configured, and enters interface configuration mode.

	Command or Action	Purpose
	`1/1`	
Step 4	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	(Optional) Sets the port to access mode only if you configured the RADIUS server.
Step 5	**no dot1x pae authenticator** **Example:** `Device(config-if)# no dot1x pae authenticator`	Disables 802.1x authentication on the port.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Resetting the 802.1x Authentication Configuration to Default Values

Beginning in privileged EXEC mode, follow these steps to reset the 802.1x authentication configuration to the default values. This procedure is optional.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet`	Enters interface configuration mode, and specifies the port to be configured.

	Command or Action	Purpose
	`1/1`	
Step 4	**dot1x default** **Example:** `Device(config-if)# dot1x default`	Resets the 802.1x parameters to the default values.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Voice-Aware 802.1x Security

You use the voice aware 802.1x security feature on the device to disable only the VLAN on which a security violation occurs, whether it is a data or voice VLAN. You can use this feature in IP phone deployments where a PC is connected to the IP phone. A security violation found on the data VLAN results in the shutdown of only the data VLAN. The traffic on the voice VLAN flows through the device without interruption.

Follow these guidelines to configure voice aware 802.1x voice security on the device:

- You enable voice aware 802.1x security by entering the **errdisable detect cause security-violation shutdown vlan** global configuration command. You disable voice aware 802.1x security by entering the **no** version of this command. This command applies to all 802.1x-configured ports in the device.

Note If you do not include the **shutdown vlan** keywords, the entire port is shut down when it enters the error-disabled state.

- If you use the **errdisable recovery cause security-violation** global configuration command to configure error-disabled recovery, the port is automatically re-enabled. If error-disabled recovery is not configured for the port, you re-enable it by using the **shutdown** and **no shutdown** interface configuration commands.
- You can re-enable individual VLANs by using the **clear errdisable interface** *interface-id* **vlan** [*vlan-list*] privileged EXEC command. If you do not specify a range, all VLANs on the port are enabled.

Follow these steps to enable voice aware 802.1x security:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**errdisable detect cause security-violation shutdown vlan** **Example:** `Device(config)# errdisable detect cause security-violation shutdown vlan`	Shut down any VLAN on which a security violation error occurs. **Note** If the **shutdown vlan** keywords are not included, the entire port enters the error-disabled state and shuts down.
Step 4	**errdisable recovery cause security-violation** **Example:** `Device(config)# errdisable recovery cause security-violation`	Enables the automatic recovery of ports that were disabled because of 802.1X security violations..
Step 5	Enter the following: • **shutdown** • **no shutdown** **Example:** `Device(config)# no shutdown`	(Optional) Re-enables an error-disabled VLAN, and clear all error-disable indications.
Step 6	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 7	**clear errdisable interface***interface-id* **vlan** *[vlan-list]* **Example:** `Device# clear errdisable interface gigabitethernet 1/1 vlan vlan_list`	(Optional) Reenables individual VLANs that have been error disabled. • For the *interface-id* argument, specify the port on which to reenable individual VLANs. • (Optional) For the *vlan-list* argument, specify a list of VLANs to be re-enabled. If vlan-list is not specified, all VLANs are re-enabled.

	Command or Action	Purpose
Step 8	**show errdisable detect** **Example:** `Device# show errdisable detect`	Displays the error-disabled detection status.

Configuration Examples for IEEE 802.1x Port-Based Authentication

The following sections provide configuration examples for IEEE 802.1x port-based authentication.

Example: Configuring Inaccessible Authentication Bypass

This example shows how to configure the Inaccessible Authentication Bypass feature:

```
Device> enable
Device# configure terminal
Device(config)# radius-server dead-criteria time 30 tries 20
Device(config)# radius-server deadtime 60
Device(config)# radius server server1
Device(config-radius-server)# address ipv4 172.29.36.49 acct-port 1618 auth-port 1612
Device(config-radius-server)# key abc1234
Device(config-radius-server)# exit
Device(config)# dot1x critical eapol
Device(config)# dot1x critical recovery delay 2000
Device(config)# interface gigabitethernet 1/1
Device(config-if)# dot1x critical
Device(config-if)# dot1x critical recovery action reinitialize
Device(config-if)# dot1x critical vlan 20
Device(config-if)# end
```

Example: Configuring VLAN Groups

This example shows how to configure VLAN groups, map VLANs to groups, and verify VLAN group configurations and mappings to specified VLANs:

```
Device> enable
Device(config)# vlan group eng-dept vlan-list 10
Device(config)# exit
Device# show vlan group group-name eng-dept

Group Name                       Vlans Mapped
-------------                    --------------
eng-dept                         10

Device# show dot1x vlan-group all

Group Name                       Vlans Mapped
-------------                    --------------
eng-dept                         10
```

```
hr-dept                          20
```

This example shows how to add a VLAN to an existing VLAN group and to verify that the VLAN is added:

```
Device> enable
Device(config)# vlan group eng-dept vlan-list 30
Device(config)# exit
Device(config)# show vlan group eng-dept

Group Name                       Vlans Mapped
-------------                    --------------
eng-dept                         10,30
```

This example shows how to remove a VLAN from a VLAN group:

```
Device> enable
Device# no vlan group eng-dept vlan-list 10
```

This example shows that when all the VLANs are cleared from a VLAN group, the VLAN group is cleared:

```
Device> enable
Device(config)# no vlan group eng-dept vlan-list 30
Vlan 30 is successfully cleared from vlan group eng-dept.
Device(config)# exit
Device# show vlan group group-name eng-dept
```

This example shows how to clear all the VLAN groups:

```
Device> enable
Device(config)# no vlan group end-dept vlan-list all
Device(config)# exit
Device# show vlan-group all
```

Monitoring IEEE 802.1x Port-Based Authentication Statistics and Status

This section list the commands to monitor IEEE 802.1x port-based authentication statistics and status.

Table 110: Privileged EXEC show Commands

Command	Purpose
show dot1x all statistics	Displays 802.1x statistics for all ports
show dot1x interface *interface-id* **statistics**	Displays 802.1x statistics for a specific port
show dot1x all [**count** | **details** | **statistics** | **summary**]	Displays the 802.1x administrative and operational status for a switch
show dot1x interface *interface-id*	Displays the 802.1x administrative and operational status for a specific port

Table 111: Global Configuration Commands

Command	Purpose
no dot1x logging verbose	Filters verbose 802.1x authentication messages.

CHAPTER 101

Web-Based Authentication

This chapter describes how to configure web-based authentication on the device. It contains these sections:

Restrictions for Web-Based Authentication

A device without host switch virtual interface (SVI) does not intercept TCP SYN packets for Cisco Identity Services Engine (ISE) posture redirection.

Information About Web-Based Authentication

Web-Based Authentication Overview

Use the web-based authentication feature, known as web authentication proxy, to authenticate end users on host systems that do not run the IEEE 802.1x supplicant.

When you initiate an HTTP session, web-based authentication intercepts ingress HTTP packets from the host and sends an HTML login page to the users. The users enter their credentials, which the web-based authentication feature sends to the authentication, authorization, and accounting (AAA) server for authentication.

If authentication succeeds, web-based authentication sends a Login-Successful HTML page to the host and applies the access policies returned by the AAA server.

If authentication fails, web-based authentication forwards a Login-Fail HTML page to the user, prompting the user to retry the login. If the user exceeds the maximum number of attempts, web-based authentication forwards a Login-Expired HTML page to the host, and the user is placed on a watch list for a waiting period.

Note HTTPS traffic interception for central web authentication redirect is not supported.

Note You should use global parameter-map (for method-type, custom, and redirect) only for using the same web authentication methods like consent, web consent, and webauth, for all the clients and SSIDs. This ensures that all the clients have the same web-authentication method.

If the requirement is to use Consent for one SSID and Web-authentication for another SSID, then you should use two named parameter-maps. You should configure Consent in first parameter-map and configure webauth in second parameter-map.

Note The traceback that you receive when webauth client tries to do authentication does not have any performance or behavioral impact. It happens rarely when the context for which FFM replied back to EPM for ACL application is already dequeued (possibly due to timer expiry) and the session becomes 'unauthorized'.

Based on where the web pages are hosted, the local web authention can be categorozied as follows:

- *Internal*—The internal default HTML pages (Login, Success, Fail, and Expire) in the controller are used during the local web authentication.
- *Customized*—The customized web pages (Login, Success, Fail, and Expire) are downloaded onto the controller and used during the local web authentication.
- *External*—The customized web pages are hosted on the external web server instead of using the in-built or custom web pages.

Based on the various web authentication pages, the types of web authentication are as follows:

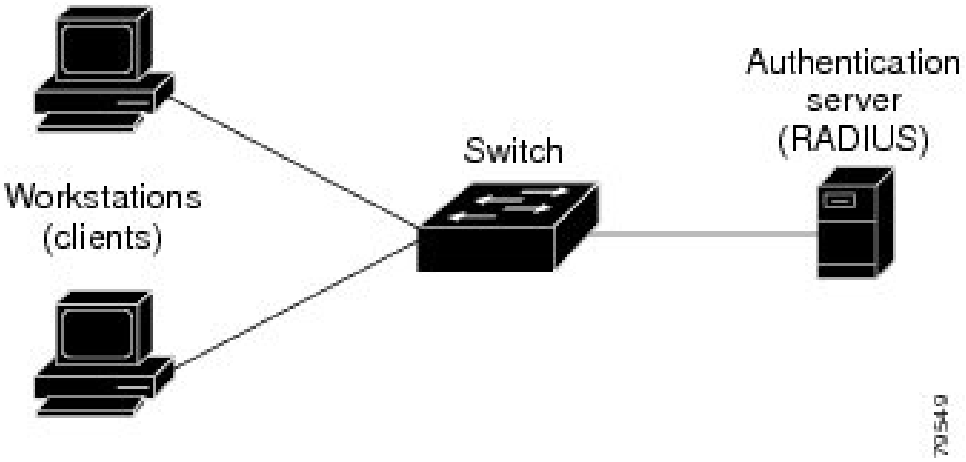
- *Webauth*—This is a basic web authentication. Herein, the controller presents a policy page with the user name and password. You need to enter the correct credentials to access the network.
- *Consent or web-passthrough*—Herein, the controller presents a policy page with the Accept or Deny buttons. You need to click the Accept button to access the network.
- *Webconsent*—This is a combination of webauth and consent web authentication types. Herein, the controller presents a policy page with Accept or Deny buttons along with user name or password. You need to enter the correct credentials and click the Accept button to access the network.

Device Roles

With web-based authentication, the devices in the network have these specific roles:

- *Client*—The device (workstation) that requests access to the LAN and the services and responds to requests from the switch. The workstation must be running an HTML browser with Java Script enabled.
- *Authentication server*—Authenticates the client. The authentication server validates the identity of the client and notifies the switch that the client is authorized to access the LAN and the switch services or that the client is denied.
- *Switch*—Controls the physical access to the network based on the authentication status of the client. The switch acts as an intermediary (proxy) between the client and the authentication server, requesting identity information from the client, verifying that information with the authentication server, and relaying a response to the client.

Figure 113: Web-Based Authentication Device Roles



Host Detection

The switch maintains an IP device tracking table to store information about detected hosts.

For Layer 2 interfaces, web-based authentication detects IP hosts by using these mechanisms:

- ARP based trigger—ARP redirect ACL allows web-based authentication to detect hosts with a static IP address or a dynamic IP address.
- Dynamic ARP inspection
- DHCP snooping—Web-based authentication is notified when the switch creates a DHCP-binding entry for the host.

Session Creation

When web-based authentication detects a new host, it creates a session as follows:

- Reviews the exception list.

  If the host IP is included in the exception list, the policy from the exception list entry is applied, and the session is established.

- Reviews for authorization bypass

  If the host IP is not on the exception list, web-based authentication sends a nonresponsive-host (NRH) request to the server.

  If the server response is access accepted, authorization is bypassed for this host. The session is established.

- Sets up the HTTP intercept ACL

  If the server response to the NRH request is access rejected, the HTTP intercept ACL is activated, and the session waits for HTTP traffic from the host.

Authentication Process

When you enable web-based authentication, these events occur:

- The user initiates an HTTP session.
- The HTTP traffic is intercepted, and authorization is initiated. The switch sends the login page to the user. The user enters a username and password, and the switch sends the entries to the authentication server.
- If the authentication succeeds, the switch downloads and activates the user's access policy from the authentication server. The login success page is sent to the user.
- If the authentication fails, the switch sends the login fail page. The user retries the login. If the maximum number of attempts fails, the switch sends the login expired page, and the host is placed in a watch list. After the watch list times out, the user can retry the authentication process.
- If the authentication server does not respond to the switch, and if an AAA fail policy is configured, the switch applies the failure access policy to the host. The login success page is sent to the user.
- The switch reauthenticates a client when the host does not respond to an ARP probe on a Layer 2 interface, or when the host does not send any traffic within the idle timeout on a Layer 3 interface.
- The switch reauthenticates a client when the host does not respond to an ARP probe on a Layer 2 interface.
- The feature applies the downloaded timeout or the locally configured session timeout.
- If the terminate action is RADIUS, the feature sends a nonresponsive host (NRH) request to the server. The terminate action is included in the response from the server.
- If the terminate action is default, the session is dismantled, and the applied policy is removed.

Local Web Authentication Banner

With Web Authentication, you can create a default and customized web-browser banners that appears when you log in to a switch.

The banner appears on both the login page and the authentication-result pop-up pages. The default banner messages are as follows:

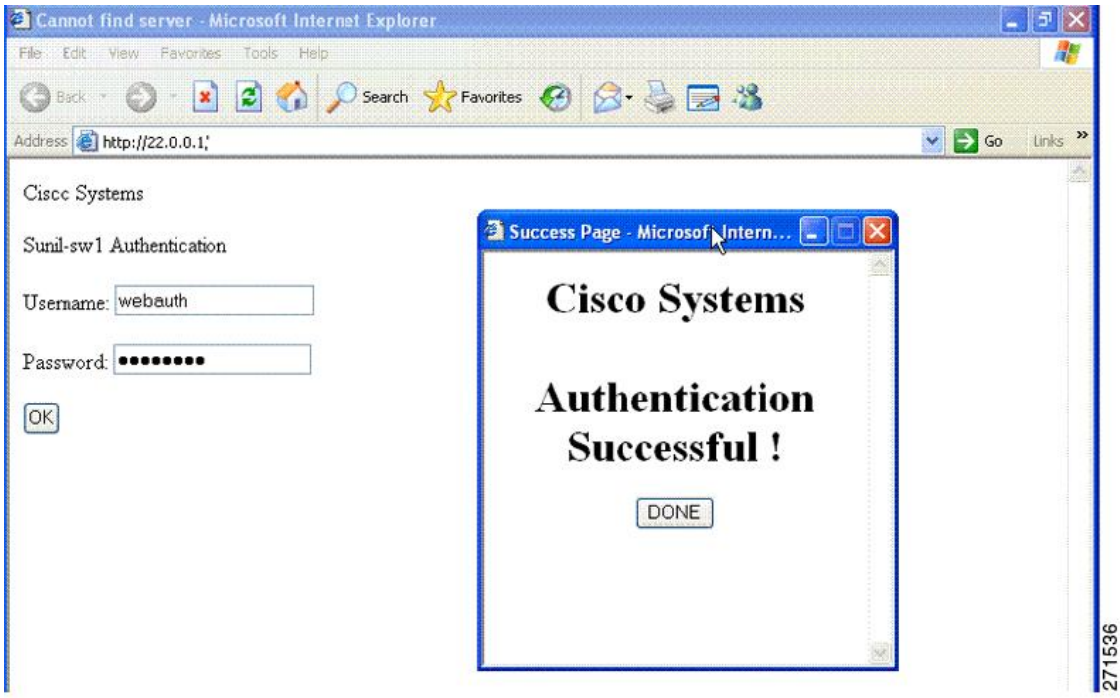
- *Authentication Successful*
- *Authentication Failed*
- *Authentication Expired*

The Local Web Authentication Banner can be configured in as follows:

- Legacy mode—Use the **ip admission auth-proxy-banner http** global configuration command.
- New-style mode—Use the **parameter-map type webauth global banner** global configuration command.

The default banner *Cisco Systems* and *Switch host-name Authentication* appear on the Login Page. *Cisco Systems* appears on the authentication result pop-up page.

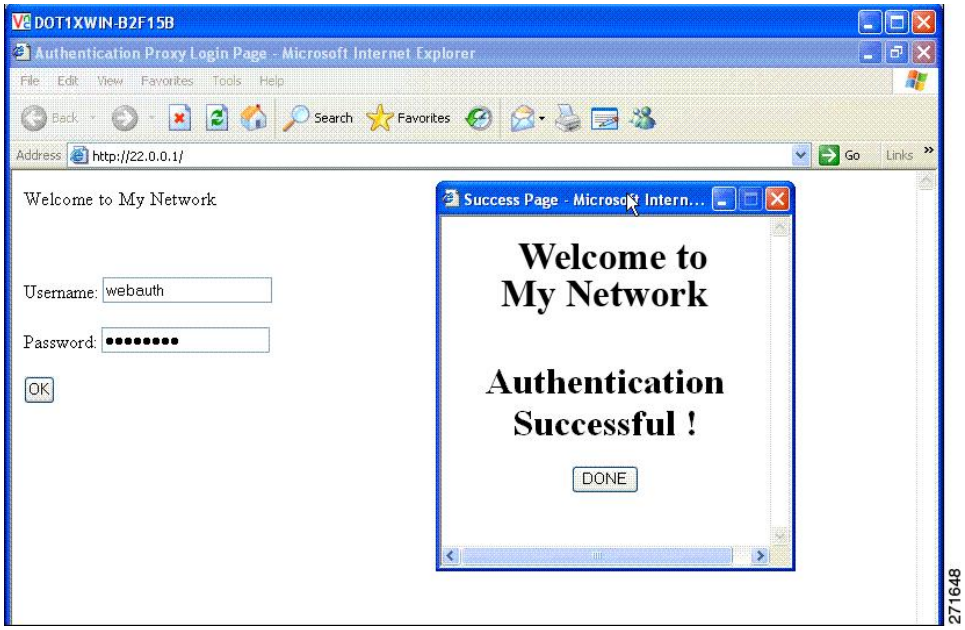
Figure 114: Authentication Successful Banner



The banner can be customized as follows:

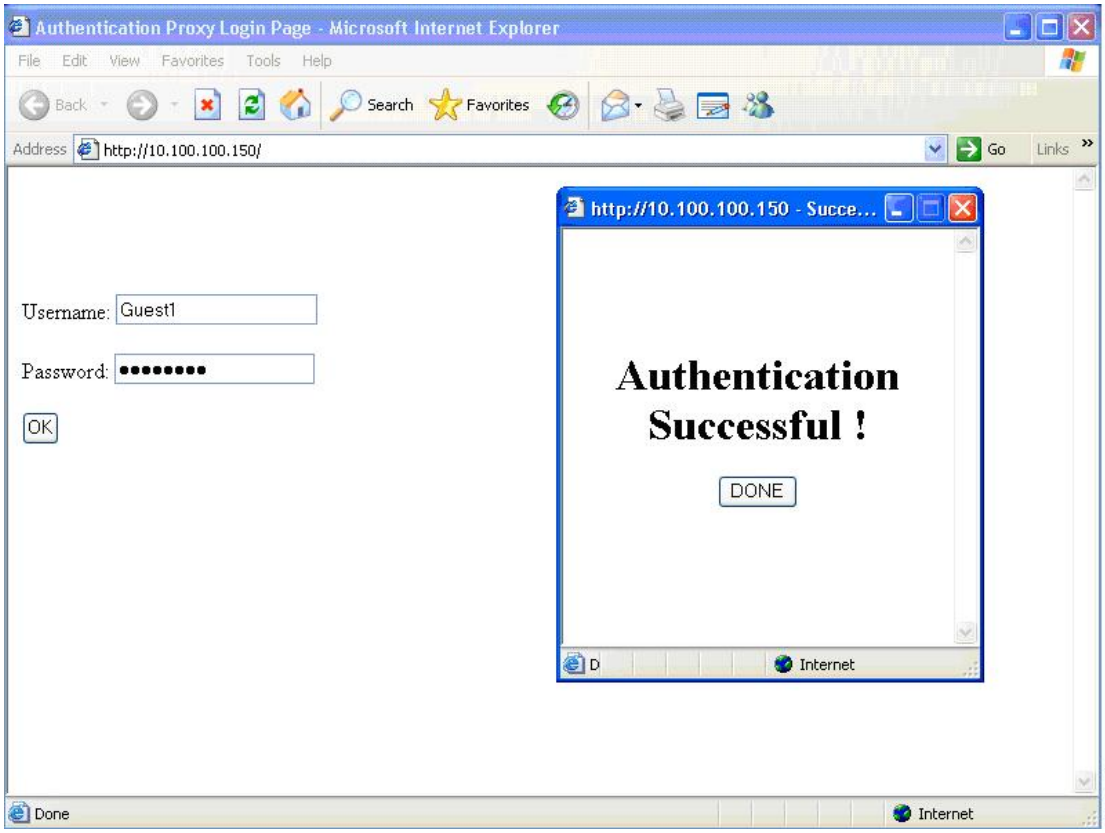
- Add a message, such as switch, router, or company name to the banner:
  - Legacy mode—Use the **ip admission auth-proxy-banner http** *banner-text*global configuration command.
  - New-style mode—Use the **parameter-map type webauth global banner** global configuration command.
- Add a logo or text file to the banner:
  - Legacy mode—Use the **ip admission auth-proxy-banner http** *file-path* global configuration command.
  - New-style mode—Use the **parameter-map type webauth global banner** global configuration command.

Figure 115: Customized Web Banner



If you do not enable a banner, only the username and password dialog boxes appear in the web authentication login screen, and no banner appears when you log into the switch.

Figure 116: Login Screen With No Banner



Web Authentication Customizable Web Pages

During the web-based authentication process, the switch internal HTTP server hosts four HTML pages to deliver to an authenticating client. The server uses these pages to notify you of these four-authentication process states:

- Login—Your credentials are requested.
- Success—The login was successful.
- Fail—The login failed.
- Expire—The login session has expired because of excessive login failures.

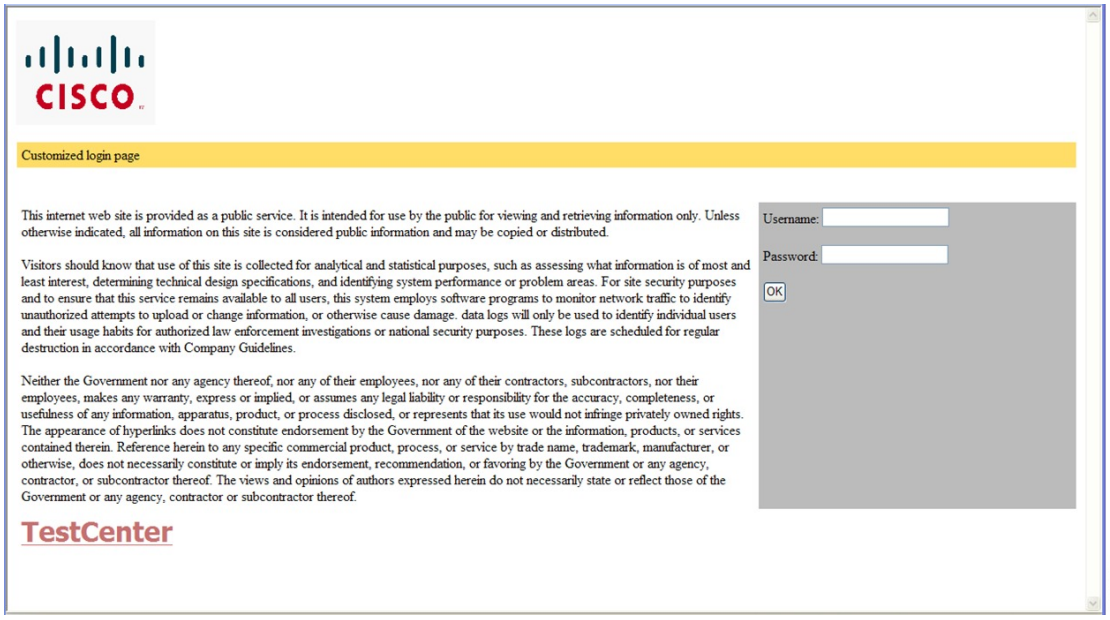
Guidelines

- You can substitute your own HTML pages for the default internal HTML pages.
- You can use a logo or specify text in the *login*, *success*, *failure*, and *expire* web pages.
- On the banner page, you can specify text in the login page.
- The pages are in HTML.
- You must include an HTML redirect command in the success page to access a specific URL.

- The URL string must be a valid URL (for example, http://www.cisco.com). An incomplete URL might cause *page not found* or similar errors on a web browser.
- If you configure web pages for HTTP authentication, they must include the appropriate HTML commands (for example, to set the page time out, to set a hidden password, or to confirm that the same page is not submitted twice). The custom page samples in the webauth bundle are provided with the image and the details of what you can and cannot change.
- The CLI command to redirect users to a specific URL is not available when the configured login form is enabled. The administrator should ensure that the redirection is configured in the web page.
- If the CLI command redirecting users to specific URL after authentication occurs is entered and then the command configuring web pages is entered, the CLI command redirecting users to a specific URL does not take effect.
- Configured web pages can be copied to the switch boot flash or flash.
- On stackable switches, configured pages can be accessed from the flash on the active switch or member switches.
- The login page can be on one flash, and the success and failure pages can be another flash (for example, the flash on the active switch or a member switch).
- You must configure all four pages.
- All of the logo files (image, flash, audio, video, and so on) that are stored in the system directory (for example, flash, disk0, or disk) and that are displayed on the login page must use *web_auth_<filename>* as the file name.
- The configured authentication proxy feature supports both HTTP and SSL.

You can substitute your HTML pages for the default internal HTML pages. You can also specify a URL to which users are redirected after authentication occurs, which replaces the internal Success page.

Figure 117: Customizable Authentication Page



Authentication Proxy Web Page Guidelines

When configuring customized authentication proxy web pages, follow these guidelines:

- To enable the custom web pages feature, specify all four custom HTML files. If you specify fewer than four files, the internal default HTML pages are used.
- The four custom HTML files must be present on the flash memory of the switch. The maximum size of each HTML file is 8 KB.
- Any images on the custom pages must be on an accessible HTTP server. Configure an intercept ACL within the admission rule.
- Any external link from a custom page requires configuration of an intercept ACL within the admission rule.
- To access a valid DNS server, any name resolution required for external links or images requires configuration of an intercept ACL within the admission rule.
- If the custom web pages feature is enabled, a configured auth-proxy-banner is not used.
- If the custom web pages feature is enabled, the redirection URL for successful login feature is not available.
- To remove the specification of a custom file, use the **no** form of the command.

Because the custom login page is a public web form, consider these guidelines for the page:

- The login form must accept user entries for the username and password and must show them as **uname** and **pwd**.
- The custom login page should follow best practices for a web form, such as page timeout, hidden password, and prevention of redundant submissions.

Redirection URL for Successful Login Guidelines

When configuring a redirection URL for successful login, consider these guidelines:

- If the custom authentication proxy web pages feature is enabled, the redirection URL feature is disabled and is not available in the CLI. You can perform redirection in the custom-login success page.
- If the redirection URL feature is enabled, a configured auth-proxy-banner is not used
- To remove the specification of a redirection URL, use the **no** form of the command.
- If the redirection URL is required after the web-based authentication client is successfully authenticated, then the URL string must start with a valid URL (for example, http://) followed by the URL information. If only the URL is given without http://, then the redirection URL on successful authentication might cause page not found or similar errors on a web browser.

Web-based Authentication Interactions with Other Features

Port Security

You can configure web-based authentication and port security on the same port. Web-based authentication authenticates the port, and port security manages network access for all MAC addresses, including that of the client. You can then limit the number or group of clients that can access the network through the port.

LAN Port IP

You can configure LAN port IP (LPIP) and Layer 2 web-based authentication on the same port. The host is authenticated by using web-based authentication first, followed by LPIP posture validation. The LPIP host policy overrides the web-based authentication host policy.

If the web-based authentication idle timer expires, the NAC policy is removed. The host is authenticated, and posture is validated again.

Gateway IP

You cannot configure Gateway IP (GWIP) on a Layer 3 VLAN interface if web-based authentication is configured on any of the switch ports in the VLAN.

You can configure web-based authentication on the same Layer 3 interface as Gateway IP. The host policies for both features are applied in software. The GWIP policy overrides the web-based authentication host policy.

ACLs

If you configure a VLAN ACL or a Cisco IOS ACL on an interface, the ACL is applied to the host traffic only after the web-based authentication host policy is applied.

For Layer 2 web-based authentication, it is more secure, though not required, to configure a port ACL (PACL) as the default access policy for ingress traffic from hosts connected to the port. After authentication, the web-based authentication host policy overrides the PACL. The Policy ACL is applied to the session even if there is no ACL configured on the port.

You cannot configure a MAC ACL and web-based authentication on the same interface.

You cannot configure web-based authentication on a port whose access VLAN is configured for VACL capture.

EtherChannel

You can configure web-based authentication on a Layer 2 EtherChannel interface. The web-based authentication configuration applies to all member channels.

How to Configure Web-Based Authentication

Default Web-Based Authentication Configuration

Table 112: Default Web-based Authentication Configuration

Feature	Default Setting
AAA	Disabled
RADIUS server • IP address • UDP authentication port • Key	• None specified • None specified
Default value of inactivity timeout	3600 seconds
Inactivity timeout	Enabled

Web-Based Authentication Configuration Guidelines and Restrictions

- Web-based authentication is an ingress-only feature.
- You can configure web-based authentication only on access ports. Web-based authentication is not supported on trunk ports, EtherChannel member ports, or dynamic trunk ports.
- External web authentication, where the switch redirects a client to a particular host or web server for displaying login message, is not supported.
- You cannot authenticate hosts on Layer 2 interfaces with static ARP cache assignment. These hosts are not detected by the web-based authentication feature because they do not send ARP messages.
- By default, the IP device tracking feature is disabled on a switch. You must enable the IP device tracking feature to use web-based authentication.
- You must enable SISF-Based device tracking to use web-based authentication. By default, SISF-Based device tracking is disabled on a switch.
- You must configure at least one IP address to run the switch HTTP server. You must also configure routes to reach each host IP address. The HTTP server sends the HTTP login page to the host.
- Hosts that are more than one hop away might experience traffic disruption if an STP topology change results in the host traffic arriving on a different port. This occurs because the ARP and DHCP updates might not be sent after a Layer 2 (STP) topology change.
- Web-based authentication does not support VLAN assignment as a downloadable-host policy.
- Web-based authentication supports IPv6 in Session-aware policy mode. IPv6 Web-authentication requires at least one IPv6 address configured on the switch and IPv6 Snooping configured on the switchport.

- Web-based authentication and Network Edge Access Topology (NEAT) are mutually exclusive. You cannot use web-based authentication when NEAT is enabled on an interface, and you cannot use NEAT when web-based authentication is running on an interface.
- Identify the following RADIUS security server settings that will be used while configuring switch-to-RADIUS-server communication:
  - Host name
  - Host IP address
  - Host name and specific UDP port numbers
  - IP address and specific UDP port numbers

  The combination of the IP address and UDP port number creates a unique identifier, that enables RADIUS requests to be sent to multiple UDP ports on a server at the same IP address. If two different host entries on the same RADIUS server are configured for the same service (for example, authentication) the second host entry that is configured functions as the failover backup to the first one. The RADIUS host entries are chosen in the order that they were configured.
- When you configure the RADIUS server parameters:
  - Specify the **key** *string* on a separate command line.
  - For **key** *string*, specify the authentication and encryption key used between the switch and the RADIUS daemon running on the RADIUS server. The key is a text string that must match the encryption key used on the RADIUS server.
  - When you specify the **key** *string*, use spaces within and at the end of the key. If you use spaces in the key, do not enclose the key in quotation marks unless the quotation marks are part of the key. This key must match the encryption used on the RADIUS daemon.
  - You can globally configure the timeout, retransmission, and encryption key values for all RADIUS servers by using with the **radius-server** global configuration command. If you want to configure these options on a per-server basis, use the **radius-server timeout**, **radius-server transmit**, and the **radius-server key** global configuration commands.

Note You need to configure some settings on the RADIUS server, including: the switch IP address, the key string to be shared by both the server and the switch, and the downloadable ACL (DACL). For more information, see the RADIUS server documentation.

- For a URL redirect ACL:
  - Packets that match a permit access control entry (ACE) rule are sent to the CPU for forwarding to the AAA server.
  - Packets that match a deny ACE rule are forwarded through the switch.
  - Packets that match neither the permit ACE rule or deny ACE rule are processed by the next dACL, and if there is no dACL, the packets hit the implicit-deny ACL and are dropped.

Configuring the Authentication Rule and Interfaces

Follow these steps to configure the authentication rule and interfaces:

Before you begin

SISF-Based device tracking is a prerequisite to Web Authentication. Ensure that you have enabled device tracking programmatically or manually.

For more information, see *Confguring SISF-Based Tracking*.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip admission name** *name* **proxy http** **Example:** `Device(config)# ip admission name webauth1 proxy http`	Configures an authentication rule for web-based authorization.
Step 4	**interface** *type slot/port* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode and specifies the ingress Layer 2 or Layer 3 interface to be enabled for web-based authentication.
Step 5	**ip access-group** *name* **Example:** `Device(config-if)# ip access-group webauthag`	Applies the default ACL.
Step 6	**ip admission name** **Example:**	Configures an authentication rule for web-based authorization for the interface.

	Command or Action	Purpose
	`Device(config)# ip admission name`	
Step 7	**exit** **Example:** `Device# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 8	**show ip admission** **Example:** `Device# show ip admission`	Displays the network admission cache entries and information about web authentication sessions.

Configuring AAA Authentication

If a method-list is configured under VTY lines, the corresponding method list must be added to the AAA configuration:

```
Device(config)# line vty 0 4
Device(config-line)# authorization commands 15 list1
Device(config-line)# exit
Device(config)# aaa authorization commands 15 list1 group tacacs+
```

If a method-list is not configured under VTY lines, you must add the default method list to the AAA configuration:

```
Device(config)# line vty 0 4
Device(config-line)# exit
Device(config)# aaa authorization commands 15 default group tacacs+
```

Follow these steps to configure AAA authentication:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA functionality.
Step 4	**aaa authentication login default group** {**tacacs+** | **radius**} **Example:** `Device(config)# aaa authentication login default group tacacs+`	Defines the list of authentication methods at login. **named_authentication_list** refers to any name that is not greater than 31 characters. **AAA_group_name** refers to the server group name. You need to define the server-group **server_name** at the beginning itself.
Step 5	**aaa authorization auth-proxy default group** {**tacacs+** | **radius**} **Example:** `Device(config)# aaa authorization auth-proxy default group tacacs+`	Creates an authorization method list for web-based authorization.
Step 6	**tacacs server** *server-name* **Example:** `Device(config)# tacacs server yourserver`	Specifies an AAA server.
Step 7	**address** {**ipv4 | ipv6**} *ip address* **Example:** `Device(config-server-tacacs)# address ipv4 10.0.1.12`	Configures the IP address for the TACACS server.
Step 8	**key** *string* **Example:** `Device(config-server-tacacs)# key cisco123`	Configures the authorization and encryption key used between the switch and the TACACS server.
Step 9	**end** **Example:** `Device(config-server-tacacs)# end`	Exits the TACACS server mode and returns to privileged EXEC mode.

Configuring Switch-to-RADIUS-Server Communication

Follow these steps to configure the RADIUS server parameters:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip radius source-interface vlan** *vlan interface number* **Example:** `Device(config)# ip radius source-interface vlan 80`	Specifies that the RADIUS packets have the IP address of the indicated interface.
Step 4	**radius server** *server name* **Example:** `Device(config)# radius server rsim address ipv4 124.2.2.12`	(Optional) Specifies the IP address of the RADIUS server.
Step 5	**address** {**ipv4** | **ipv6**} *ip address* **Example:** `Device(config-radius-server)# address ipv4 10.0.1.2 auth-port 1550 acct-port 1560`	Configures the IP address for the RADIUS server.
Step 6	**key** *string* **Example:** `Device(config-radius-server)# key rad123`	(Optional) Specifies the authentication and encryption key used between the switch and the RADIUS daemon running on the RADIUS server.

	Command or Action	Purpose
Step 7	**exit** **Example:** `Device(config-radius-server)# exit`	Exits the RADIUS server mode and enters the global configuration mode.
Step 8	**radius-server vsa send authentication** *string* **Example:** `Device(config)# radius-server vsa send authentication`	Enable downloading of an ACL from the RADIUS server.
Step 9	**radius-server dead-criteria** [**time** *seconds*] [**tries** *num-tries*] **Example:** `Device(config)# radius-server dead-criteria tries 45`	Configures the conditions that determine when a RADIUS server is considered unavailable or dead. Enter **time** in seconds during which there is no response from RADIUS server to the device. Enter number of **tries** where there will be no valid response from RADIUS server to the device. The range of *num-tries* is 1 to 100.
Step 10	**end** **Example:** `Device# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring the HTTP Server

To use web-based authentication, you must enable the HTTP server within the device. You can enable the server for either HTTP or HTTPS.

Note The Apple psuedo-browser will not open if you configure only the **ip http secure-server** command. You should also configure the **ip http server** command.

Follow the procedure given below to enable the server for either HTTP or HTTPS:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip http server** **Example:** `Device(config)# ip http server`	Enables the HTTP server. The web-based authentication feature uses the HTTP server to communicate with the hosts for user authentication.
Step 4	**ip http secure-server** **Example:** `Device(config)# ip http secure-server`	Enables HTTPS. You can configure custom authentication proxy web pages or specify a redirection URL for successful login. **Note** To ensure secure authentication when you enter the **ip http secure-server** command, the login page is always in HTTPS (secure HTTP) even if the user sends an HTTP request.
Step 5	**end** **Example:** `Device# end`	Exits global configuration mode and returns to privileged EXEC mode.

Customizing the Authentication Proxy Web Pages

You can configure web authentication to display four substitute HTML pages to the user in place of the default HTML pages during web-based authentication.

Follow these steps to specify the use of your custom authentication proxy web pages:

Before you begin

Store your custom HTML files on the device flash memory.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip admission proxy http login page file** *device:login-filename* **Example:** `Device(config)# ip admission proxy http login page file disk1:login.htm`	Specifies the location in the device memory file system of the custom HTML file to use in place of the default login page. The *device:* is flash memory.
Step 4	**ip admission proxy http success page file** *device:success-filename* **Example:** `Device(config)# ip admission proxy http success page file disk1:success.htm`	Specifies the location of the custom HTML file to use in place of the default login success page.
Step 5	**ip admission proxy http failure page file** *device:fail-filename* **Example:** `Device(config)# ip admission proxy http fail page file disk1:fail.htm`	Specifies the location of the custom HTML file to use in place of the default login failure page.
Step 6	**ip admission proxy http login expired page file** *device:expired-filename* **Example:** `Device(config)# ip admission proxy http login expired page file disk1:expired.htm`	Specifies the location of the custom HTML file to use in place of the default login expired page.
Step 7	**end** **Example:** `Device# end`	Exits global configuration mode and returns to privileged EXEC mode.

Specifying a Redirection URL for a Successful Login

Follow these steps to specify a URL to which the user is redirected after authentication, effectively replacing the internal Success HTML page:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip admission proxy http success redirect** *url-string* **Example:** `Device(config)# ip admission proxy http success redirect www.example.com`	Specifies a URL for redirection of the user in place of the default login success page.
Step 4	**end** **Example:** `Device# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Web-Based Authentication Parameters

Follow these steps to configure the maximum number of failed login attempts before the client is placed in a watch list for a waiting period:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip admission max-login-attempts** *number* **Example:** `Device(config)# ip admission max-login-attempts 10`	Sets the maximum number of failed login attempts. The range is 1 to 2147483647 attempts. The default is 5.
Step 4	**exit** **Example:** `Device# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring a Web-Based Authentication Local Banner

Follow these steps to configure a local banner on a switch that has web authentication configured.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip admission auth-proxy-banner http** [*banner-text* | *file-path*] **Example:** `Device(config)# ip admission auth-proxy-banner http C My Switch C`	Enables the local banner. (Optional) Create a custom banner by entering *C banner-text C* (where *C* is a delimiting character), or *file-path* that indicates a file (for example, a logo or text file) that appears in the banner.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device# end`	Exits global configuration mode and returns to privileged EXEC mode.

Removing Web-Based Authentication Cache Entries

Follow these steps to remove web-based authentication cache entries:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ip auth-proxy cache** {* | *host ip address*} **Example:** `Device# clear ip auth-proxy cache 192.168.4.5`	Delete authentication proxy entries. Use an asterisk to delete all cache entries. Enter a specific IP address to delete the entry for a single host.
Step 3	**clear ip admission cache** {* | *host ip address*} **Example:** `# clear ip admission cache 192.168.4.5`	Delete authentication proxy entries. Use an asterisk to delete all cache entries. Enter a specific IP address to delete the entry for a single host.

Verifying Web-Based Authentication

Use the commands in this topic to display the web-based authentication settings for all interfaces or for specific ports.

Table 113: Privileged EXEC show Commands

Command	Purpose
show authentication sessions method webauth	Displays the web-based authentication settings for the interface.

Command	Purpose
show authentication sessions interface *type slot/port*[**details**]	Displays the web-based authentication settings for the specified interface . In Session Aware Networking mode, use the **show authentication sessions interface** command.

CHAPTER 102

Identity Based Networking Services Overview

Cisco Identity Based Networking Services (IBNS) provides a policy and identity-based framework in which edge devices can deliver flexible and scalable services to subscribers. This module provides information about what Cisco IBNS is and its features and benefits.

Cisco Identity Based Networking Services Overview

Cisco IBNS provides a policy and identity based framework in which edge devices can deliver flexible and scalable services to subscribers. This module provides information about what Cisco IBNS is and its features and benefits.

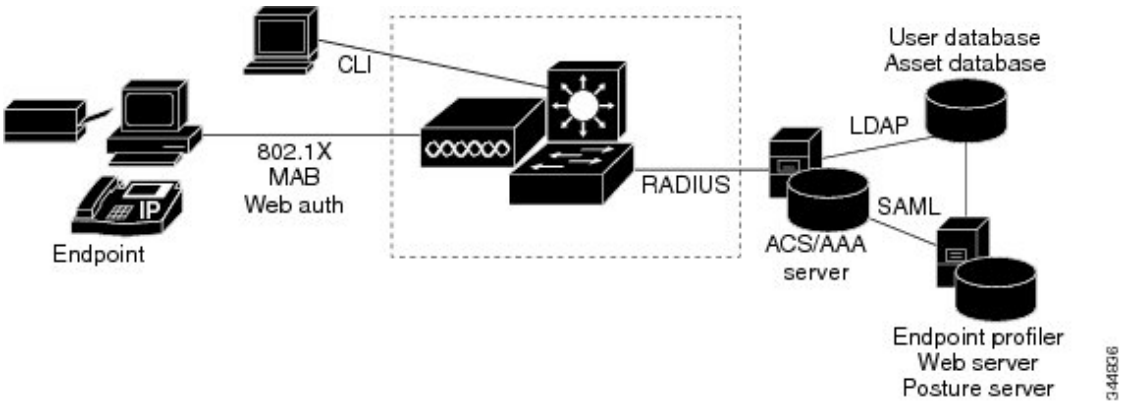
Information About Identity-Based Networking Services

Understanding Cisco Identity Based Networking Services

Cisco IBNS feature provides a policy and identity-based framework in which edge devices can deliver flexible and scalable services to subscribers. Cisco IBNS provides an identity-based approach to access management and subscriber management. It offers a consistent way to configure features across technologies, a command interface that allows easy deployment and customization of features, and a robust policy control engine with the ability to apply policies defined locally or received from an external server to enforce policy in the network.

The figure below illustrates a typical deployment of Cisco IBNS in a physically distributed enterprise with a campus, branch offices, and remote workers.

Figure 118: Sample Deployment of Cisco IBNS



Features in Cisco Identity Based Networking Services

Cisco IBNS includes the following features:

- Cisco common classification policy language (C3PL)-based identity configuration
- Concurrent authentication methods on a single session, including IEEE 802.1x (dot1x), MAC authentication bypass (MAB), and web authentication
- Downloadable identity service templates
- Extended RADIUS change of authorization (CoA) support for querying, reauthenticating, and terminating a session, port shutdown and port bounce, and activating and deactivating an identity service template.
- Local authentication using Lightweight Directory Access Protocol (LDAP)
- Locally defined identity control policies
- Locally defined identity service templates
- Per-user inactivity handling across methods
- Web authentication support of common session ID
- Web authentication support of IPv6

Benefits of Cisco Identity Based Networking Services

Identity-based solutions are essential for delivering access control for disparate groups such as employees, contractors, and partners while maintaining low operating expenses. Cisco IBNS provides a consistent approach to operational management through a policy and identity-based infrastructure leading to faster deployment of new features and easier management of switches.

Cisco IBNS provides the following benefits:

- An identity-based framework for session management.
- A robust policy control engine to apply policies defined locally or received from an external AAA server.
- Faster deployment and customization of features across access technologies.

- A simpler and consistent way to configure features across access methods, platforms, and application domains.

Web Authentication Support for Common Session ID

Cisco IBNS allows a single session identifier to be used for web authentication sessions in addition to all 802.1X and MAB authenticated sessions for a client. This session ID is used for all reporting purposes such as show commands, MIBs, and RADIUS messages and allows users to distinguish messages for one session from messages for other sessions. This common session ID is used consistently across all authentication methods and features applied to a session.

Web Authentication Support of IPv6

Cisco IBNS introduces IPv6 support for web authentication. IPv6 is supported for web authentication only when Cisco IBNS is explicitly configured. This means that you must permanently convert your configuration to the Cisco common classification policy language (C3PL) display mode by specifically configuring a Cisco IBNS command such as the **policy-map type control subscriber** command.

IP Device Tracking

IP device tracking can be configured using the Switch Integrated Security Features (SISF) policy. Use the tracking enable command in device tracking configuration mode, to configure device tracking using SISF policy. Use the **show device-tracking** command to display the device tracking configuration.

The following is the sample configuration for device tracking.

```
Device(config)# device-tracking policy sisf_policy
Device(config-device-tracking)# tracking enable
Device(config-device-tracking)# exit
Device(config)# interface GigabitEthernet 1/1
Device(config-if)# switchport mode access
Device(config-if)# device-tracking attach-policy sisf_policy
Device(config-if)# end
```

CHAPTER 103

Change of Authorization Support

Authentication provides a method to identify users, which includes the login and password dialog, challenge and response, messaging support, and encryption, depending on the selected security protocol. Authentication is the way a user is identified prior to being allowed access to the network and network services.

Change of Authorization Support

Cisco Identity Based Networking Services (IBNS) supports RADIUS change of authorization (CoA) commands for session query, reauthentication, and termination, port bounce and port shutdown, and service template activation and deactivation. This module provides information about the supported CoA commands for Cisco IBNS.

Information About CoA Support

RADIUS Change-of-Authorization Support

Cisco IOS software supports the RADIUS CoA extensions defined in RFC 5176 that are typically used in a push model to allow the dynamic reconfiguring of sessions from external AAA or policy servers. Per-session CoA requests are supported for session identification, session termination, host reauthentication, port shutdown, and port bounce. This model comprises one request (CoA-Request) and two possible response codes:

- CoA acknowledgement (ACK) [CoA-ACK]
- CoA nonacknowledgement (NAK) [CoA-NAK]

The request is initiated from a CoA client (typically a AAA or policy server) and directed to the device that acts as a listener.

The table below shows the RADIUS CoA commands and vendor-specific attributes (VSAs) supported by Cisco IBNS. All CoA commands must include the session identifier between the device and the CoA client.

Table 114: RADIUS CoA Commands Supported by Cisco IBNS

CoA Command	Cisco VSA
Activate service	Cisco:Avpair="subscriber:command=activate-service" Cisco:Avpair="subscriber:service-name=<*service-name*>" Cisco:Avpair="subscriber:precedence=<*precedence-number*>" Cisco:Avpair="subscriber:activation-mode=replace-all"
Deactivate service	Cisco:Avpair="subscriber:command=deactivate-service" Cisco:Avpair="subscriber:service-name=<*service-name*>"
Bounce host port	Cisco:Avpair="subscriber:command=bounce-host-port"
Disable host port	Cisco:Avpair="subscriber:command=disable-host-port"
Session query	Cisco:Avpair="subscriber:command=session-query"
Session reauthenticate	Cisco:Avpair="subscriber:command=reauthenticate" Cisco:Avpair="subscriber:reauthenticate-type=last" or Cisco:Avpair="subscriber:reauthenticate-type=rerun"
Session terminate	This is a standard disconnect request and does not require a VSA.
Interface template	Cisco:AVpair="interface-template-name=<*interfacetemplate*>"

Session Identification

For disconnect and CoA requests targeted at a particular session, the device locates the session based on one or more of the following attributes:

- Acct-Session-Id (IETF attribute #44)
- Audit-Session-Id (Cisco VSA)
- Calling-Station-Id (IETF attribute #31, which contains the host MAC address)
- IPv6 Attributes, which can be one of the following:
  - Framed-IPv6-Prefix (IETF attribute #97) and Framed-Interface-Id (IETF attribute #96), which together create a full IPv6 address per RFC 3162
  - Framed-IPv6-Address
- Plain IP Address (IETF attribute #8)

If more than one session identification attribute is included in the message, all of the attributes must match the session or the device returns a Disconnect-NAK or CoA-NAK with the error code Invalid Attribute Value.

For CoA requests targeted at a particular enforcement policy, the device returns a CoA-NAK with the error code Invalid Attribute Value if any of the above session identification attributes are included in the message.

CoA Activate Service Command

The CoA activate service command can be used to activate a service template on a session. The AAA server sends the request in a standard CoA-Request message using the following VSAs:

Cisco:Avpair="subscriber:command=activate-service"

Cisco:Avpair="subscriber:service-name=<*service-name*>"

Cisco:Avpair="subscriber:precedence=<*precedence-number*>"

Cisco:Avpair="subscriber:activation-mode=replace-all"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the Session Identification, on page 1530 section. If the device cannot locate a session, it returns a CoA-NAK message with the Session Context Not Found error-code attribute. If the device locates a session, it initiates an activate template operation for the hosting port and a CoA-ACK is returned. If activating the template fails, a CoA-NAK message is returned with the Error-Code attribute set to the appropriate message.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

CoA Deactivate Service Command

The CoA deactivate service command can be used to deactivate a service template on a session. The AAA server sends the request in a standard CoA-Request message using the following VSAs:

Cisco:Avpair="subscriber:command=deactivate-service"

Cisco:Avpair="subscriber:service-name=<*service-name*>"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the Session Identification, on page 1530 section. If the device cannot locate a session, it returns a CoA-NAK message with the Session Context Not Found error-code attribute. If the device locates a session, it initiates a deactivate template operation for the hosting port and a CoA-ACK is returned. If deactivating the template fails, a CoA-NAK message is returned with the Error-Code attribute set to the appropriate message.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

CoA Bounce Host Port Command

The CoA bounce host port command terminates a session and bounces the port (initiates a link down event followed by a link up event). The AAA server sends the request in a standard CoA-Request message with the following VSA:

Cisco:Avpair="subscriber:command=bounce-host-port"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the Session Identification, on page 1530 section. If the session cannot be located, the device returns a CoA-NAK message with the Session Context Not Found error-code attribute. If the session is located, the device disables the hosting port for a period of ten seconds, reenables it (port bounce), and returns a CoA-ACK.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

The CoA bounce port command is useful as a last resort when an endpoint needs to acquire a new IP address after a change in authorization and this is the only way to indicate to the endpoint to restart the DHCP process. This can occur when there is a VLAN change and the endpoint is a device, such as a printer, that does not have a mechanism to detect a change on this authentication port. This command can cause a link flap on an authentication port, which triggers DHCP renegotiation from one or more hosts connected to this port.

CoA Disable Host Port Command

The CoA disable host port command administratively shuts down the authentication port that is hosting a session, which terminates the session. The AAA server sends the request in a standard CoA-Request message with the following VSA:

Cisco:Avpair="subscriber:command=disable-host-port"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the Session Identification, on page 1530 section. If the device cannot locate the session, it returns a CoA-NAK message with the Session Context Not Found error-code attribute. If the device locates the session, it disables the hosting port and returns a CoA-ACK message.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

CoA Session Query Command

The CoA session query command requests service information about a subscriber session. The AAA server sends the request in a standard CoA-Request message containing the following VSA:

Cisco:Avpair="subscriber:command=session-query"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the Session Identification, on page 1530 section. If the device cannot locate a session, it returns a CoA-NAK message with the Session Context Not Found error-code attribute. If the device locates a session, it performs a session query operation on the session and returns a CoA-ACK message.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

CoA Session Reauthenticate Command

To initiate session authentication, the AAA server sends a standard CoA-Request message containing the following VSAs:

Cisco:Avpair="subscriber:command=reauthenticate"

Cisco:Avpair="subscriber:reauthenticate-type=<last | rerun>"

"reauthenticate-type" defines whether the CoA reauthentication request uses the authentication method that last succeeded on the session or whether the authentication process is completely rerun.

The following rules apply:

- "subscriber:command=reauthenticate" must be present to trigger a reauthentication.

- If “subscriber:reauthenticate-type” is not specified, the default behavior is to rerun the last successful authentication method for the session. If the method reauthenticates successfully, all old authorization data is replaced with the new reauthenticated authorization data.
- “subscriber:reauthenticate-type” is valid only when included with “subscriber:command=reauthenticate.” If it is included in another CoA command, the VSA will be silently ignored.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is resent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

CoA Session Terminate Command

A CoA Disconnect-Request command terminates a session without disabling the host port. This command causes reinitialization of the authenticator state machine for the specified host, but does not restrict the host’s access to the network. If the session cannot be located, the device returns a Disconnect-NAK message with the Session Context Not Found error-code attribute. If the session is located, the device terminates the session. After the session has been completely removed, the device returns a Disconnect-ACK.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client.

To restrict a host’s access to the network, use a CoA Request with the Cisco:Avpair=“subscriber:command=disable-host-port” VSA. This command is useful when a host is known to cause problems on the network and network access needs to be immediately blocked for the host. When you want to restore network access on the port, reenable it using a non-RADIUS mechanism.

CHAPTER 104

Configuring Identity Control Policies

Configuring Identity Control Policies

Identity control policies define the actions that Cisco Identity Based Networking Services (IBNS) takes in response to specified conditions and subscriber events. A variety of system actions, conditions, and events can be combined using a consistent policy language. This module provides information about how to configure identity control policies for Cisco IBNS.

Information About Identity Control Policies

Cisco Identity Based Networking Services Configuration

To convert all relevant authentication commands to their Class-Based Policy Language(CPL) control policy equivalents, use the **authentication convert-to new-style** command. This command permanently converts the legacy configuration on the switch to identity-based networking services.

Note This configuration is irreversible. It disables the conversion command – **authentication display [legacy | new-style]**.

Use the **authentication display config-mode** command in EXEC mode to display the current configuration mode; *legacy* if it is legacy mode and **new-style** if it is Identity-Based Networking Services configuration mode.

```
(Device)# authentication display config-mode
Current configuration mode is legacy

Device)# authentication display config-mode
Current configuration mode is new-style
```

Concurrent Authentication Methods

Cisco IBNS allows the concurrent operation of IEEE 802.1x (dot1x), MAC authentication bypass (MAB), and web authentication methods, making it possible to invoke multiple authentication methods in parallel on

a single subscriber session. This allows the client-supported method to complete at the earliest opportunity without the delays associated with serialization.

Typically, the access control method that is used to authorize a host is left up to the endpoint. For example, a printer without an 802.1x supplicant would be authorized through MAB only, an employee desktop through 802.1x only, and a guest through web authentication only. The default priority order is 802.1x, followed by MAB, then web authentication. When method priorities are the same, the first method that successfully authenticates the session prevails.

An example in which more than one method may succeed during the lifetime of a session is when MAB is used to provide interim access pending success of 802.1x. A host could be also be given interim access to a web server to allow credentials to be updated so that 802.1x can succeed after an authentication failure.

Configuration Display Mode

Identity-Based Networking Services introduces new Cisco IOS commands that replace many of the previously supported authentication and policy commands. These commands are available only after enabling the Cisco common classification policy language (C3PL) display mode that supports Identity-Based Networking Services. Identity-Based Networking Services features such as concurrent authentication and web authentication with IPv6 are not supported in legacy mode.

The device defaults to the legacy configuration mode until you do one of the following:

- Enter the **authentication display new-style** command—This command switches to C3PL display mode, temporarily converting your legacy configuration to a Identity-Based Networking Services configuration so you can see how it looks before you make the conversion permanent. You can switch back to legacy mode by using the **authentication display legacy** command. See the Enabling the Display Mode for Cisco Identity Based Networking Services, on page 1538 section.
- Enter a Identity-Based Networking Services configuration command—After you enter the first explicit Identity-Based Networking Services command, the configuration converts to C3PL display mode permanently and legacy commands are suppressed. The **authentication display** command is disabled and you can no longer revert to the legacy configuration mode.

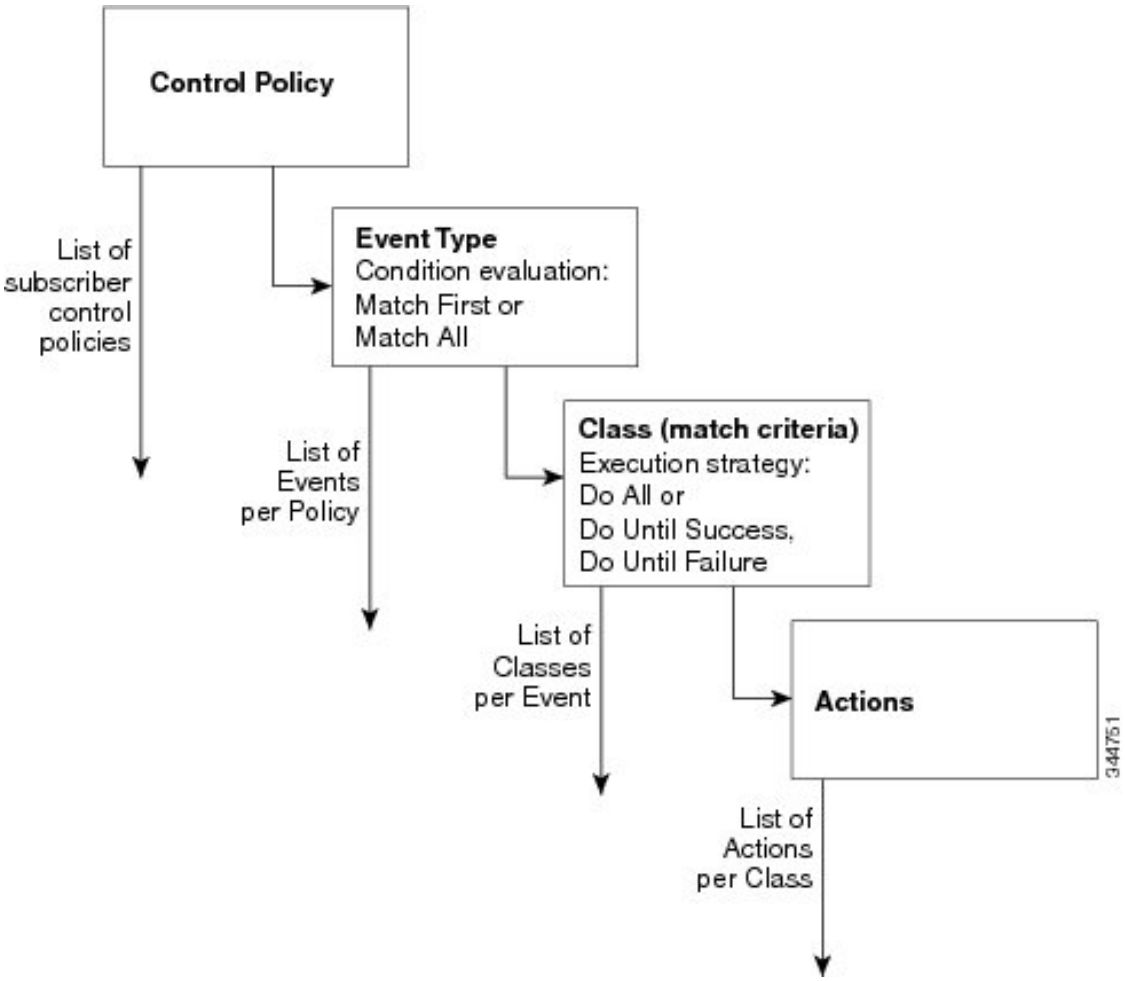
Control Policies for Cisco Identity Based Networking Services

A control policy defines the handling of different subscriber life-cycle events. For various events, such as session start or session failure, you can specify actions in the control policy. These actions can be executed conditionally for different subscribers based on various match criteria. Control policies are activated on interfaces and typically control the authentication of subscriber identity and the activation of services on sessions. For example, you can configure a control policy to authenticate specific subscribers and then provide them with access to specific services.

A control policy consists of one or more control policy rules and a decision strategy that governs how the policy rules are evaluated. A control policy rule consists of a control class (a flexible condition clause), an event for which the condition is evaluated, and one or more actions. Actions are general system functions, such as authenticate or activate. You define the specific actions that an event will trigger and some events have default actions.

The figure below illustrates how each control policy contains a list of events that are considered applicable to the subscriber life cycle. Within each event type is a list of control classes with different match criteria for subscriber identity, and under each class is a list of actions to be executed.

Figure 119: Control Policy Structure



Control Policy Configuration Overview

Control policies express system functionality in terms of an event, a condition, and an action. There are three steps in defining a control policy:

1. Create one or more control classes—A control class specifies the conditions that must be met for a control policy to be activated. A control class can contain multiple conditions, each of which will evaluate as either true or false. Match directives specify whether all, any, or none of the individual conditions must evaluate true for the class to evaluate true. Or, you can specify the default control class which does not contain any conditions and always evaluates true.
2. Create a control policy—A control policy contains one or more control policy rules. A control policy rule consists of a control class, an event that causes the class to be evaluated, and one or more actions. Actions are numbered and executed sequentially.
3. Apply the control policy—A control policy is activated by applying it to an interface.

Parameter Maps for Cisco Identity Based Networking Services

A parameter map allows you to specify parameters that control the behavior of actions specified under a control policy. For Cisco IBNS, an authentication parameter map defines parameters used for the action specified with the **authenticate using webauth** command. You can configure the following types of parameter maps:

- Authentication bypass (This is also called nonresponsive host [NRH] authentication.)
- Consent
- Web authentication
- Web authentication with consent

Parameter maps are optional. If you do not configure a named parameter map, the software uses the default parameters that are specified in the global parameter map.

Per User Inactivity Handling Across Methods

A common inactivity aging feature extends support for RADIUS attributes 28 (Idle-Timeout) and attribute 29 (Termination-Action) to web authenticated sessions, providing consistent inactivity handling across all authentication methods, including 802.1x, MAC authentication bypass (MAB), and web authentication. The AAA server sends these attributes as part of the user authorization. After a session has been idle for the amount of time specified in attribute 28, or has reached the timeout configured with attribute 29, the session is terminated.

You can also apply the inactivity timeout and absolute timeout to sessions through a locally defined service template. When enabling the inactivity timeout, you can also enable address resolution protocol (ARP) probes that are sent before the session is terminated. For configuration information, see the *Configuring IdentityService Templates* module.

How to Configure Identity Control Policies

Enabling the Display Mode for Cisco Identity Based Networking Services

Cisco IBNS features are configured in the Cisco common classification policy language (C3PL) display mode. The legacy authentication manager mode is enabled by default. You can use the following procedure to switch to C3PL display mode and temporarily convert any legacy configuration commands to their C3PL equivalents. This allows you to preview your legacy configuration as a Identity-Based Networking Services configuration before making the conversion permanent. After you enter an explicit Cisco IBNS command, the conversion becomes permanent and you can no longer revert to legacy mode.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**authentication display** {**legacy** | **new-style**} **Example:** `Device# authentication display new-style`	Sets the display mode for authentication and policy configuration. • The default display mode is legacy. • You can use this command to switch between legacy and C3PL display mode until you execute the first explicit Identity-Based Networking Services command. After you enter the first explicit Identity-Based Networking Services command, for example when configuring a control class or control policy, the system displays a prompt to confirm whether you want to continue because this command will be disabled and you cannot revert to legacy mode. **Note** If you save the configuration while the new-style mode is enabled, and then perform a reload, the display mode is permanently set to new-style. The **authentication display** command is disabled and you cannot revert to legacy mode. For standalone devices to revert to legacy mode, save the new-style configuration in a flash, **write erase** the device and then perform a **reload**.

Configuring a Control Class

A control class defines the conditions under which the actions of a control policy are executed. You define whether all, any, or none of the conditions must evaluate true to execute the actions of the control policy. Control classes are evaluated based on the event specified in the control policy.

Note This procedure shows all of the match conditions that you can configure in a control class. You must specify at least one condition in a control class to make it valid. All other conditions, and their corresponding steps, are optional (steps 4 through 18 below).

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map type control subscriber** {**match-all** | **match-any** | **match-none**} *control-class-name* **Example:** `Device(config)# class-map type control subscriber match-all DOT1X_NO_AGENT`	Creates a control class and enters control class-map filter mode. • **match-all**—All of the conditions in the control class must evaluate true. • **match-any**—At least one of the conditions in the control class must evaluate true. • **match-none**—All of the conditions in the control class must evaluate false.
Step 4	{**match** | **no-match**} **activated-service-template** *template-name* **Example:** `Device(config-filter-control-classmap)# match activated-service-template SVC_1`	(Optional) Creates a condition that evaluates true based on the service template activated on a session.
Step 5	{**match** | **no-match**} **authorization-status** {**authorized** | **unauthorized**} **Example:** `Device(config-filter-control-classmap)# match authorization-status authorized`	(Optional) Creates a condition that evaluates true based on a session's authorization status.
Step 6	{**match** | **no-match**} **authorizing-method-priority** {**eq** | **gt** | **lt**} *priority-value* **Example:** `Device(config-filter-control-classmap)# match authorizing-method-priority eq 10`	(Optional) Creates a condition that evaluates true based on the priority of the authorization method. • **eq**—Current priority is equal to *priority-value*. • **gt**—Current priority is greater than *priority-value*. • **lt**—Current priority is less than *priority-value*. • *priority-value*—Priority value to match. Range: 1 to 254, where 1 is the highest priority and 254 is the lowest.
Step 7	{**match** | **no-match**} **client-type** {**data** | **switch** | **video** | **voice**} **Example:**	(Optional) Creates a condition that evaluates true based on an event's device type.

	Command or Action	Purpose
	`Device(config-filter-control-classmap)# match client-type data`	
Step 8	{**match** | **no-match**} **current-method-priority** {**eq** | **gt** | **lt**} *priority-value* **Example:** `Device(config-filter-control-classmap)# match current-method-priority eq 10`	(Optional) Creates a condition that evaluates true based on the priority of the current authentication method.
Step 9	{**match** | **no-match**} **ip-address** *ip-address* **Example:** `Device(config-filter-control-classmap)# match ip-address 10.10.10.1`	(Optional) Creates a condition that evaluates true based on an event's source IPv4 address.
Step 10	{**match** | **no-match**} **ipv6-address** *ipv6-address* **Example:** `Device(config-filter-control-classmap)# match ipv6-address FE80::1`	(Optional) Creates a condition that evaluates true based on an event's source IPv6 address.
Step 11	{**match** | **no-match**} **mac-address** *mac-address* **Example:** `Device(config-filter-control-classmap)# match mac-address aabb.cc00.6500`	(Optional) Creates a condition that evaluates true based on an event's MAC address.
Step 12	{**match** | **no-match**} **method** {**dot1x** | **mab** | **webauth**} **Example:** `Device(config-filter-control-classmap)# match method dot1x`	(Optional) Creates a condition that evaluates true based on an event's authentication method.
Step 13	{**match** | **no-match**} **port-type** {**l2-port** | **l3-port** | **dot11-port**} **Example:** `Device(config-filter-control-classmap)# match port-type l2-port`	(Optional) Creates a condition that evaluates true based on an event's interface type.
Step 14	{**match** | **no-match**} **result-type** [**method** {**dot1x** | **mab** | **webauth**}] *result-type* **Example:** `Device(config-filter-control-classmap)# match result-type agent-not-found`	(Optional) Creates a condition that evaluates true based on the specified authentication result. • To display the available result types, use the question mark (**?**) online help function.

	Command or Action	Purpose
Step 15	{**match** | **no-match**} **service-template** *template-name* **Example:** `Device(config-filter-control-classmap)# match service-template svc_1`	(Optional) Creates a condition that evaluates true based on an event's service template.
Step 16	{**match** | **no-match**} **tag** *tag-name* **Example:** `Device(config-filter-control-classmap)# match tag tag_1`	(Optional) Creates a condition that evaluates true based on the tag associated with an event.
Step 17	{**match** | **no-match**} **timer** *timer-name* **Example:** `Device(config-filter-control-classmap)# match timer restart`	(Optional) Creates a condition that evaluates true based on an event's timer.
Step 18	{**match** | **no-match**} **username** *username* **Example:** `Device(config-filter-control-classmap)# match username josmiths`	(Optional) Creates a condition that evaluates true based on an event's username.
Step 19	**end** **Example:** `Device(config-filter-control-classmap)# end`	(Optional) Exits control class-map filter configuration mode and returns to privileged EXEC mode.
Step 20	**show class-map type control subscriber** {**all** | **name** *control-class-name*} **Example:** `Device# show class-map type control subscriber all`	(Optional) Displays information about Identity-Based Networking Services control classes.

Example: Control Class

The following example shows a control class that is configured with two match conditions:

```
class-map type control subscriber match-all DOT1X_NO_AGENT
 match method dot1x
 match result-type agent-not-found
```

Configuring a Control Policy

Control policies determine the actions that the system takes in response to specified events and conditions. The control policy contains one or more control policy rules that associate a control class with one or more actions. The actions that you can configure in a policy rule depend on the type of event that you specify.

Note This task includes all of the actions that you can configure in a control policy regardless of the event. All of these actions, and their corresponding steps, are optional (steps 6 through 21 below). To display the supported actions for a particular event, use the question mark (**?**) online help function.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type control subscriber** *control-policy-name* **Example:** `Device(config)# policy-map type control subscriber POLICY_1`	Defines a control policy for subscriber sessions.
Step 4	**event** *event-name* [**match-all** | **match-first**] **Example:** `Device(config-event-control-policymap)# event session-started`	Specifies the type of event that triggers actions in a control policy if conditions are met. • **match-all** is the default behavior. • To display the available event types, use the question mark (**?**) online help function. For a complete description of event types, see the **event** command.
Step 5	*priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**] **Example:** `Device(config-class-control-policymap)# 10 class always`	Associates a control class with one or more actions in a control policy. • A named control class must first be configured before specifying it with the *control-class-name* argument. • **do-until-failure** is the default behavior.
Step 6	*action-number* **activate** {**policy type control subscriber** *control-policy-name* [**child** [**no-propagation** | **concurrent**] | **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** *number*] [**replace-all**]} **Example:**	(Optional) Activates a control policy or service template on a subscriber session.

	Command or Action	Purpose
	`Device(config-action-control-policymap)# 10 activate service-template FALLBACK`	
Step 7	*action-number* **authenticate using** {**dot1x** | **mab** | **webauth**} [**aaa** {**authc-list** *authc-list-name* | **authz-list** *authz-list-name*]} [**merge**] [**parameter-map** *map-name*] [**priority** *priority-number*] [**replace** | **replace-all**] [**retries** *number* {**retry-time** *seconds*}] **Example:** `Device(config-action-control-policymap)# 10 authenticate using dot1x priority 10`	(Optional) Initiates the authentication of a subscriber session using the specified method.
Step 8	*action-number* **authentication-restart** *seconds* **Example:** `Device(config-action-control-policymap)# 20 authentication-restart 60`	(Optional) Sets a timer to restart the authentication process after an authentication or authorization failure.
Step 9	*action-number* **authorize** **Example:** `Device(config-action-control-policymap)# 10 authorize`	(Optional) Initiates the authorization of a subscriber session.
Step 10	*action-number* **clear-authenticated-data-hosts-on-port** **Example:** `Device(config-action-control-policymap)# 20 clear-authenticated-data-hosts-on-port`	(Optional) Clears authenticated data hosts on a port after an authentication failure.
Step 11	*action-number* **clear-session** **Example:** `Device(config-action-control-policymap)# 30 clear-session`	(Optional) Clears an active subscriber session.
Step 12	*action-number* **deactivate** {**policy type control subscriber** *control-policy-name* | **service-template** *template-name*} **Example:** `Device(config-action-control-policymap)# 20 deactivate service-template interface_template`	(Optional) Deactivates a control policy or service template on a subscriber session.
Step 13	*action-number* **err-disable** **Example:** `Device(config-action-control-policymap)# 10 err-disable`	(Optional)Temporarily disables a port after a session violation event.

	Command or Action	Purpose
Step 14	*action-number* **pause reauthentication** **Example:** `Device(config-action-control-policymap)# 20 pause reauthentication`	(Optional) Pauses reauthentication after an authentication failure.
Step 15	*action-number* **protect** **Example:** `Device(config-action-control-policymap)# 10 protect`	(Optional) Silently drops violating packets after a session violation event.
Step 16	*action-number* **replace** **Example:** `Device(config-action-control-policymap)# 10 replace`	(Optional) Clears the existing session and creates a new session after a violation event.
Step 17	*action-number* **restrict** **Example:** `Device(config-action-control-policymap)# 10 restrict`	(Optional) Drops violating packets and generates a syslog entry after a session violation event.
Step 18	*action-number* **resume reauthentication** **Example:** `Device(config-action-control-policymap)# 20 resume reauthentication`	(Optional) Resumes the reauthentication process after an authentication failure.
Step 19	*action-number* **set-timer** *timer-name seconds* **Example:** `Device(config-action-control-policymap)# 20 set-timer RESTART 60`	(Optional) Starts a named policy timer.
Step 20	*action-number* **terminate** {**dot1x** | **mab** | **webauth**} **Example:** `Device(config-action-control-policymap)# 20 terminate webauth`	(Optional) Terminates an authentication method on a subscriber session.
Step 21	*action-number* **unauthorize** **Example:** `Device(config-action-control-policymap)# 20 unauthorize`	(Optional) Removes all authorization data from a subscriber session.
Step 22	**end** **Example:** `Device(config-action-control-policymap)# end`	(Optional) Exits control policy-map action configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 23	**show policy-map type control subscriber** {**all** | **name** *control-policy-name*} **Example:** `Device# show policy-map type control subscriber POLICY_1`	(Optional) Displays information about identity control policies.

Example: Control Policy

The following example shows a simple control policy with the minimum configuration necessary for initiating authentication:

```
policy-map type control subscriber POLICY_1
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using dot1x
```

For detailed examples of control policies for concurrent and sequential authentication, see the Configuration Examples for Cisco Identity-Based Control Policies, on page 1551 section.

Applying a Control Policy to an Interface

Control policies typically control the authentication of subscriber identity and the activation of services on sessions. Perform this task to apply a control policy to an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies an interface and enters interface configuration mode.
Step 4	**service-policy type control subscriber** *control-policy-name* **Example:** `Device(config-if)# service-policy type control subscriber POLICY_1`	Applies a previously configured control policy. • To display a list of all configured control policies, use the question mark (**?**) online help function.

	Command or Action	Purpose
Step 5	**subscriber aging** {**inactivity-timer** *seconds* [**probe**] | **probe**} **Example:** `Device(config-if)# subscriber aging inactivity-timer 60 probe`	Enables an inactivity timer for subscriber sessions. If you configure this command, you must also configure the device-tracking binding reachable-lifetime command in global configuration mode, for probes to work as expected. Configure a reachable lifetime with the same value as the inactivity timer probe. This way, when the reachable lifetime expires, the state of the entry changes based on the reachability of the host. For more information, see the device-tracking binding command in the command reference of the corresponding release.

Example: Applying a Control Policy to an Interface

```
interface GigabitEthernet 1/1
 subscriber aging inactivity-timer 60 probe
 service-policy type control subscriber POLICY_1
 subscriber aging inactivity-timer 60 probe
```

Configuring Authentication Features on Ports

Perform this task to control access to a port, including the port authorization state, host access mode, preauthentication access, and the authentication direction.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode for the selected interface.
Step 4	**access-session port-control** {**auto** | **force-authorized** | **force-unauthorized**}	Sets the authorization state of a port. • The default value is **force-authorized**.

	Command or Action	Purpose
	Example: `Device(config-if)# access-session port-control auto`	
Step 5	**access-session host-mode** {**multi-auth** | **multi-domain** | **multi-host** | **single-host**} **Example:** `Device(config-if)# access-session host-mode single-host`	Allows hosts to gain access to a controlled port. • To use this command, you must first enable the **access-session port-control auto** command. • The default value is **multi-auth**.
Step 6	**access-session closed** **Example:** `Device(config-if)# access-session closed`	Prevents preauthentication access on this port. • The port is set to open access by default.
Step 7	**access-session control-direction** {**both** | **in**} **Example:** `Device(config-if)# access-session control-direction in`	Sets the direction of authentication control on a port. • The default value is **both**.
Step 8	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 9	**show access-session interface** *interface-type interface-number* [**details**] **Example:** `Device# show access-session interface gigabitethernet 1/1 details`	Displays information about subscriber sessions that match the specified client interface.

Example: Port Authentication

```
interface GigabitEthernet 1/1
 access-session host-mode single-host
 access-session closed
 access-session port-control auto
 access-session control-direction in
```

Configuring a Parameter Map for Web-Based Authentication

A parameter map allows you to modify parameters that control the behavior of actions configured under a control policy. A parameter map for web-based authentication sets parameters that can be applied to subscriber sessions during authentication. If you do not create a parameter map, the policy uses default parameters.

Perform the following steps to define either a global or named parameter map for web-based authentication.

Note The configuration commands available in the global parameter map differ from the commands available in a named parameter map.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**parameter-map type webauth** {*parameter-map-name* | **global**} **Example:** `Device(config)# parameter-map type webauth MAP_2`	Creates a parameter map and enters parameter-map webauth configuration mode. • The specific configuration commands supported for a global parameter map defined with the **global** keyword differ from the commands supported for a named parameter map defined with the *parameter-map-name* argument.
Step 4	**banner** {**file** *location*:*filename* | **text** *banner-text*} **Example:** `Device(config-params-parameter-map)# banner file flash:webauth_banner.html`	(Optional) Displays a banner on the web-authentication login web page.
Step 5	**consent** **Example:** `Device(config-params-parameter-map)# type consent`	(Optional) Defines the methods supported by a web-based authentication parameter map. • This command is supported in named parameter maps only.
Step 6	**consent email** **Example:** `Device(config-params-parameter-map)# consent email`	(Optional) Requests a user's e-mail address on the web-authentication login web page. • This command is supported in named parameter maps only.
Step 7	**custom-page** {**failure** | **login** [**expired**] | **success**} **device** *location*:*filename* **Example:**	(Optional) Displays custom authentication proxy web pages during web-based authentication.

	Command or Action	Purpose
	`Device(config-params-parameter-map)# custom-page login device flash:webauth_login.html` `Device(config-params-parameter-map)# custom-page login expired device flash:webauth_expire.html` `Device(config-params-parameter-map)# custom-page success device flash:webauth_success.html` `Device(config-params-parameter-map)# custom-page failure device flash:webauth_fail.html`	• You must configure all four custom HTML files. If fewer than four files are configured, the internal default HTML pages will be used.
Step 8	**max-http-conns** *number* **Example:** `Device(config-params-parameter-map)# max-http-conns 5`	(Optional) Limits the number of HTTP connections for each web authentication client.
Step 9	**redirect** {{**for-login** | **on-failure** | **on-success**} *url* | **portal** {**ipv4** *ipv4-address* | **ipv6** *ipv6-address*}} **Example:** `Device(config-params-parameter-map)# redirect portal ipv6 FE80::1` `Device(config-params-parameter-map)# redirect on-failure http://10.10.3.34/~sample/failure.html`	(Optional) Redirects users to a particular URL during web-based authentication.
Step 10	**timeout init-state sec** *seconds* **Example:** `Device(config-params-parameter-map)# timeout init-state sec 60`	(Optional) Sets the Init state timeout for web-based authentication sessions. • The range of seconds is (60-3932100).
Step 11	**type** {**authbypass** | **consent** | **webauth** | **webconsent**} **Example:** `Device(config-params-parameter-map)# type consent`	(Optional) Defines the methods supported by a web-based authentication parameter map. • This command is supported in named parameter maps only.
Step 12	**virtual-ip** {**ipv4** *ipv4-address* | **ipv6** *ipv6-address*} **Example:** `Device(config-params-parameter-map)# virtual-ip ipv6 FE80::1`	(Optional) Specifies a virtual IP address for web-based authentication clients. • This command is supported in the global parameter map only.
Step 13	**watch-list** {**add-item** {**ipv4** *ipv4-address* | **ipv6** *ipv6-address*} | **dynamic-expiry-timeout** *minutes* | **enabled**} **Example:** `Device(config-params-parameter-map)# watch-list enabled`	(Optional) Enables a watch list of web-based authentication clients. • This command is supported in the global parameter map only.

	Command or Action	Purpose
	`Device(config-params-parameter-map)# watch-list dynamic-expiry-timeout 20` `Device(config-params-parameter-map)# watch-list add-item ipv6 FE80::1`	
Step 14	**end** **Example:** `Device(config-params-parameter-map)# end`	(Optional) Exits parameter-map configuration mode and returns to privileged EXEC mode.
Step 15	**show ip admission status** [**banners** | **custom-pages** | **parameter-map** [*parameter-map*]] **Example:** `Device# show ip admission status custom-pages`	(Optional) Displays information about configured banners and custom pages.

Example: Parameter Map for Web-Based Authentication

```
parameter-map type webauth PMAP_2
 type webconsent
 timeout init-state sec 60
 max-http-conns 5
 type consent
 consent email
 custom-page login device flash:webauth_login.html
 custom-page success device flash:webauth_success.html
 custom-page failure device flash:webauth_fail.html
 custom-page login expired device flash:webauth_expire.html
```

What to do next

Apply the parameter map to sessions by specifying it in the **authenticate using** command when configuring a Control Policy. See the Configuring a Control Policy, on page 1542 section.

Configuration Examples for Cisco Identity-Based Control Policies

Example: Configuring Control Policy for Concurrent Authentication Methods

The following example shows a control policy that is configured to allow concurrent authentication. All three methods (dot1x, MAB, and web authentication) are run simultaneously when a session is started. The dot1x method is set to the highest priority and web authentication has the lowest priority, which means that if multiple methods succeed, the highest priority method is honored.

If authentication fails, the session manager checks whether all methods have failed, and if so, it sets the restart timer to 60 seconds, after which it attempts to start all three methods again. On authentication success, the session manager terminates any lower priority methods; for dot1x, this is MAB and webauth; for MAB it is webauth. Lastly, if session manager detects a dot1x client (agent-found) it triggers only dot1x to run.

The class map named ALL-FAILED checks that all three methods have run to completion (result type is none until then) and that none of them was successful. In other words, all three methods have completed and failed.

Note When configuring a control policy for concurrent authentication, you must include a policy rule that explicitly terminates one method after another method of a higher priority succeeds.

```
class-map type control subscriber match-all ALL_FAILED
 no-match result-type method dot1x none
 no-match result-type method dot1x success
 no-match result-type method mab none
 no-match result-type method mab success
 no-match result-type method webauth none
 no-match result-type method webauth success
!
class-map type control subscriber match-all DOT1X
 match method dot1x
!
class-map type control subscriber match-all MAB
 match method mab
!
policy-map type control subscriber CONCURRENT_DOT1X_MAB_WEBAUTH
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using mab priority 20
   20 authenticate using dot1x priority 10
   30 authenticate using webauth parameter-map WEBAUTH_DEFAULT priority 30
 event authentication-failure match-first
  10 class ALL_FAILED
   10 authentication-restart 60
 event authentication-success match-all
  10 class DOT1X
   10 terminate MAB
   20 terminate webauth
  20 class MAB
   10 terminate webauth
 event agent-found match-all
  10 class always do-until-failure
   10 authenticate using dot1x priority 10
```

Example: Configuring Control Policy for Sequential Authentication Methods

The following example shows a control policy that is configured to allow sequential authentication methods using 802.1X (dot1x), MAB, and web authentication.

```
parameter-map type webauth WEBAUTH_FALLBACK
 type webauth
!
class-map type control subscriber match-all DOT1X_NO_RESP
 match method dot1x
 match result-type method dot1x agent-not-found
!
class-map type control subscriber match-all MAB_FAILED
 match method mab
 match result-type method mab authoritative
!
policy-map type control subscriber POLICY_Gi3/0/10
```

```
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using dot1x priority 10
 event authentication-failure match-first
  10 class DOT1X_NO_RESP do-until-failure
   10 terminate dot1x
   20 authenticate using mab priority 20
  20 class MAB_FAILED do-until-failure
   10 terminate mab
   20 authenticate using webauth parameter-map WEBAUTH_FALLBACK priority 30
  30 class always do-until-failure
   10 terminate dot1x
   20 terminate mab
   30 terminate webauth
   40 authentication-restart 60
 event agent-found match-all
  10 class always do-until-failure
   10 terminate mab
   20 terminate webauth
   30 authenticate using dot1x priority 10
```

The following example shows a control policy that is configured to allow sequential authentication methods using 802.1X and MAB. If authentication fails, a service template for VLAN is activated.

```
service-template VLAN210
 vlan 210
!
class-map type control subscriber match-all DOT1X_FAILED
 match method dot1x
 match result-type method dot1x authoritative
!
class-map type control subscriber match-all DOT1X_NO_RESP
 match method dot1x
 match result-type method dot1x agent-not-found
!
class-map type control subscriber match-all MAB_FAILED
 match method mab
 match result-type method mab authoritative
!
policy-map type control subscriber POLICY_Gi3/0/14
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using dot1x retries 2 retry-time 0 priority 10
 event authentication-failure match-first
  10 class DOT1X_NO_RESP do-until-failure
   10 terminate dot1x
   20 authenticate using mab priority 20
  20 class MAB_FAILED do-until-failure
   10 terminate mab
   20 activate service-template VLAN210
   30 authorize
  30 class DOT1X_FAILED do-until-failure
   10 terminate dot1x
   20 authenticate using mab priority 20
  40 class always do-until-failure
   10 terminate dot1x
   20 terminate mab
   30 authentication-restart 60
 event agent-found match-all
  10 class always do-until-failure
   10 terminate mab
   20 authenticate using dot1x retries 2 retry-time 0 priority 10
```

Example: Configuring Parameter Maps

Global Parameter Map

The following example shows the configuration of a global parameter map:

```
parameter-map type webauth global
 timeout init-state sec 15
 watch-list enabled
 virtual-ip ipv6 FE80::1
 redirect on-failure http://10.10.3.34/~sample/failure.html
 max-http-conns 100
 watch-list dynamic-expiry-timeout 5000
 banner file flash:webauth_banner.html
```

Named Parameter Maps for Web Authentication and Authentication Bypass (nonresponsive host [NRH])

The following example shows the configuration of two named parameter maps; one for web authentication and one for authentication bypass. This example also shows the corresponding control policy configuration.

```
parameter-map type webauth WEBAUTH_BANNER
 type webauth
 banner
!
parameter-map type webauth WEBAUTH_NRH
 type authbypass
!
class-map type control subscriber match-all NRH_FAIL
 match method webauth
 match current-method-priority eq 254
!
policy-map type control subscriber WEBAUTH_NRH
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using webauth parameter-map WEBAUTH_NRH priority 254
 event authentication-failure match-all
  10 class NRH_FAIL do-until-failure
   10 terminate webauth
   20 authenticate using webauth parameter-map WEBAUTH_BANNER priority 30
```

Named Parameter Map for Web Authentication Using Custom Pages

The following example shows the configuration of a named parameter map for web authentication that defines custom pages for the login process, along with a control policy that uses the parameter map.

```
parameter-map type webauth CUSTOM_WEBAUTH
 type webauth
 custom-page login device flash:login_page.htm
 custom-page success device flash:success_page.htm
 custom-page failure device flash:fail_page.htm
 custom-page login expired device flash:expire_page.htm
!
policy-map type control subscriber CUSTOM_WEBAUTH
 event session-started match-all
```

```
  10 class always do-until-failure
   10 authenticate using webauth parameter-map CUSTOM_WEB retries 2 retry-time 0
```

Named Parameter Map for Consent

The following example shows the configuration of a named parameter map for consent, along with the corresponding control policy that uses the parameter map:

```
parameter-map type webauth CONSENT
 type consent
!
ip access-list extended GUEST_ACL
 permit ip any 172.30.30.0 0.0.0.255
 permit ip any host 172.20.249.252
!
service-template GUEST_POLICY
 access-group GUEST_ACL
!
policy-map type control subscriber CONSENT
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using webauth parameter-map CONSENT
 event authentication-success match-all
  10 class always do-until-failure
   10 activate service-template GUEST_POLICY
```

Named Parameter Map for Web Authentication with Consent

The following example shows the configuration of a named parameter map for web authentication with consent, along with the corresponding control policy that uses the parameter map:

```
parameter-map type webauth WEBAUTH_CONSENT
 type webconsent
!
ip access-list extended GUEST_ACL
 permit ip any 172.30.30.0 0.0.0.255
 permit ip any host 172.20.249.252
!
service-template GUEST_POLICY
 access-group GUEST_ACL
!
policy-map type control subscriber WEBAUTH_CONSENT
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using webauth parameter-map CONSENT
 event authentication-success match-all
  10 class always do-until-failure
   10 activate service-template GUEST_POLICY
```

CHAPTER 105

Policy Classification Engine

Policy Classification Engine

The Policy Classification Engine feature helps configure device-based policies and client (network endpoint) profiling and enforces a per user or per device policy on a network. The policy classification engine enables bring-your-own-device (BYOD) deployments integrate user or wireless device policies into the wireless controller. This module explains how to configure policies and apply them to a wireless LAN (WLAN).

Restrictions for Policy Classification Engine

Interface templates are not valid on wireless sessions.

Information About Policy Classification Engine

Policy Classification Engine Overview

The Policy Classification Engine feature helps configure device-based policies and client (network endpoint) profiling and enforces a per user or per device policy on a network.

You can configure sets of different policies that can be used for lookup and sequential matching. A policy is matched based on the configured policy statement. Use policies to profile devices based on the Dynamic Host Control Protocol (DHCP) or HTTP to identify end devices in a network. You can enforce specific policies at network endpoints.

The device uses these attributes and predefined classification profiles to identify devices.

Policies are configured based on the following parameters:

- Device—Types of end devices. Examples are Windows machines, smart phones, Apple device like iPads, iPhones, and so on.
- Regular expressions
- User role—The user type or user group to which an user belongs. Examples are students, employees, and so on.
- Username—Login credentials entered by users.

- Time-of-day—The time-of-day when endpoints are allowed into a network.
- OUI—The MAC address that identifies the Organizational Unique Identifier (OUI).
- MAC address—The MAC address of the endpoint.

Once the device (switch) has a match corresponding to the policy parameters per end point, a policy is added. Policy enforcement is based on the following session attributes:

- VLAN—User-defined VLAN
- Access control list (ACL)
- Session timeout value—User-defined timeout for client sessions
- Quality of service (QoS)

You can configure policies and based on the session attributes, enforce these policies on end points.

How to Configure Policy Classification Engine

Configuring Policies in Cisco Identity Based Networking Services

To configure policies, perform the following tasks:

1. Configure a service template.

   For more information, see the *Configuring Identity Service Templates* module.
2. Configure an interface template.

   For more information, see the *About Interface Templates* module.
3. Create a parameter map.
4. Create a policy map.
5. Apply the policy on a wireless LAN (WLAN).

Configuring a Subscriber Parameter Map

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**parameter-map type subscriber attribute-to-service** *parameter-map-name* **Example:** `Device(config)# parameter-map type subscriber attribute-to-service param-map`	Configures a subscriber parameter map and enters parameter-map filter configuration mode.
Step 4	*priority-number* **map device-type eq** *device-type* **oui eq** *MAC-address* **Example:** `Device(config-parameter-map-filter)# 1 map device-type eq "Cisco-IP-Phone-9971" oui eq "08.cc.68"`	Maps the priority and the Organizationally Unique Identifier (OUI) of the configured device, and enters parameter-map filter submode configuration mode.
Step 5	*action-number* **interface-template** *interface-template-name* **Example:** `Device(config-parameter-map-filter-submode)# 2 interface-template IP-PHONE-INTERFACE-TEMPLATE`	Maps the action number to an interface template.
Step 6	**end** **Example:** `Device(config-parameter-map-filter-submode)# end`	Exits parameter-map filter submode configuration mode and returns to privileged EXEC mode.
Step 7	**show parameter-map type subscriber attribute-to-service** *parameter-map-name* **Example:** `Device# show parameter-map type subscriber attribute-to-service parameter-map-name`	Displays information about the specified parameter map.

Example

The following is sample output from the **show parameter-map type subscriber attribute-to-service** command:

```
Device# show parameter-map type subscriber attribute-to-service param-map

Parameter-map name: param-map
 Map: 1 map device-type eq "Cisco-IP-Phone-9971" oui eq "08.cc.68"
  Action(s):
   2 interface-template IP-PHONE-INTERFACE-TEMPLATE
```

Configuring a Subscriber Policy Map

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type control subscriber** *policy-map-name* **Example:** `Device(config)# policy-map type control subscriber pmap`	Defines a control policy for subscriber sessions and enters control policy-map event configuration mode.
Step 4	**event identity-update** {**match-all** | **match-first**} **Example:** `Device(config-event-control-policymap)# event identity-update match-all`	Specifies the event type that triggers actions in a control policy if conditions are met, and enters control policy-map class configuration mode.
Step 5	*priority-number* **class always** {**do-all** | **do-until-failure** | **do-until-success**} **Example:** `Device(config-class-control-policymap)# 1 class always do-until-failure`	Associates a control class with one or more actions in a control policy and enters control policy-map action configuration mode.
Step 6	*action-number* **map attribute-to-service table** *parameter-map-name* **Example:** `Device(config-action-control-policymap)# 2 map attribute-to-service table param-map`	Maps identity-update attribute to an autoconf template.
Step 7	**end** **Example:** `Device(config-action-control-policymap)# end`	Exits control policy-map action configuration mode and returns to privileged EXEC mode.
Step 8	**show policy-map type control subscriber** *policy-map-name* **Example:** `Device# show policy-map type control subscriber pmap`	Displays information and statistics about the control policies.

Example

The following is sample output from the **show policy-map type control subscriber** command:

```
Device# show policy-map type control subscriber pmap

show policy-map type control subscriber pmap
policy-map
  event identity-update match-all
    1 class always do-until-failure
      1 map attribute-to-service table param-map
```

Configuration Examples for Policy Classification Engine

Example: Configuring a Subscriber Parameter Map

```
Device# configure terminal
Device(config)# parameter-map type subscriber attribute-to-service param-map
Device(config-parameter-map-filter)# 1 map device-type eq "Cisco-IP-Phone-9971" oui "eq
08.cc.68"
Device(config-parameter-map-filter-submode)# 2 interface-template IP-PHONE-INTERFACE-TEMPLATE
Device(config-parameter-map-filter-submode)# end
```

Example: Configuring a Subscriber Policy Map

```
Device# configure terminal
Device(config)# policy-map type control subscriber pmap
Device(config-event-control-policymap)# event identity-update match-all
Device(config-class-control-policymap)# 1 class always do-until-failure
Device(config-action-control-policymap)# 2 map attribute-to-service table param-map
Device(config-action-control-policymap)# end
```

CHAPTER 106

Configuring Identity Service Templates

Configuring Identity Service Templates

Identity service templates contain a set of policy attributes or features that can be applied to one or more subscriber sessions through a control policy, a RADIUS Change of Authorization (CoA) request, or a user profile or service profile. This module provides information about how to configure local service templates for Identity-Based Networking Services.

Prerequisites for Identity Service Templates

For downloadable service templates, the switch uses the default password “cisco123” when downloading the service templates from the authentication, authorization, and accounting (AAA) server, Cisco Secure Access Control Server (ACS), or Cisco Identity Services Engine (ISE). The AAA, ACS, and ISE server must include the password “cisco123” in the service template configuration.

Information About Identity Service Templates

Service Templates for Cisco Identity-Based Networking Services

A service template contains a set of service-related attributes or features, such as access control lists (ACLs) and VLAN assignments, that can be activated on one or more subscriber sessions in response to session life-cycle events. Templates simplify the provisioning and maintenance of network session policies where policies fall into distinct groups or are role-based.

A service template is applied to sessions through its reference in a control policy, through RADIUS Change of Authorization (CoA) requests, or through a user profile or service profile. User profiles are defined per subscriber; service profiles can apply to multiple subscribers.

Identity-Based Networking Services supports two types of service templates:

- Downloadable Service Templates—The service template is configured centrally on an external ACS or AAA server and downloaded on demand.
- Locally Configured Service Templates—The service template is configured locally on the device through the Cisco IOS command-line interface (CLI).

Downloadable Service Templates

Cisco Identity Based Networking Services (IBNS) can download a service template defined on an external AAA server. The template defines a collection of AAA attributes. These templates are applied to sessions through the use of vendor-specific attributes (VSAs) included in RADIUS CoA messages received from the external AAA server or ACS. The name of the template is referenced in a user profile or a control policy, which triggers a download of the service template during processing.

The downloadable template is cached on the device and subsequent requests for a download will refer to the available cached template. The template however is cached only for the duration of its active usage. The downloaded template cached on the device is protected and cannot be deleted through the command line interface or through other applications. This ensures that the template is deleted only when there are no active references to it.

Locally Configured Service Templates

Service templates can be configured locally through the CLI. These service templates can be applied to subscriber sessions by a reference in a control policy.

When an active local template is updated, changes to that local template will be reflected across all sessions for which the template is active. If a template is deleted, all content from that template that is applied against sessions is removed.

How to Configure Identity Service Templates

Configuring a Local Service Template

A service template defines the local policies that can be applied to a subscriber session. Activate this service template on sessions on which the local policies must be applied.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service-template** *template-name* **Example:** `Device(config)# service-template SVC_2`	Creates a service template and enters service template configuration mode.
Step 4	**absolute-timer** *minutes* **Example:** `Device(config-service-template)# absolute-timer 15`	(Optional) Enables an absolute timeout for subscriber sessions.

	Command or Action	Purpose
Step 5	**access-group** *access-list-name* **Example:** `Device(config-service-template)# access-group ACL_2`	(Optional) Applies an access list to sessions using a service template.
Step 6	**description** *description* **Example:** `Device(config-service-template)# description label for SVC_2`	(Optional) Adds a description for a service template.
Step 7	**inactivity-timer** *minutes* [**probe**] **Example:** `Device(config-service-template)# inactivity-timer 15`	(Optional) Enables an inactivity timeout for subscriber sessions.
Step 8	**redirect url** *url* **Example:** `Device(config-service-template)# redirect url www.cisco.com`	(Optional) Redirects clients to a particular URL.
Step 9	**sgt** *range* **Example:** `Device(config-service-template)# sgt 100`	(Optional) Associates a Security Group Tag (SGT) with a service template.
Step 10	**tag** *tag-name* **Example:** `Device(config-service-template)# tag TAG_2`	(Optional) Associates a user-defined tag with a service template.
Step 11	**vlan** *vlan-id* **Example:** `Device(config-service-template)# vlan 215`	(Optional) Applies a VLAN to sessions using a service template.
Step 12	**end** **Example:** `Device(config-service-template)# end`	Exits service template configuration mode and returns to privileged EXEC mode.
Step 13	**show service-template** [*template-name*] **Example:** `Device# show service-template SVC_2`	Displays information about configured service templates.

Example: Service Template

```
service-template SVC_2
 description label for SVC_2
 access-group ACL_2
 redirect url www.cisco.com
 vlan 215
 inactivity-timer 15
 absolute-timer 15
 tag TAG_2
```

What to do next

To activate a service template on a subscriber session, specify the service template in a control policy. See Configuring a Control Policy.

Configuration Examples for Identity Service Templates

Example: Activating a Service Template and Replace All

Local Service Template Configuration

The following example shows the configuration of a service template defined locally on the device. This template contains attributes that are applied to sessions that use the control policy named POSTURE_VALIDATION, shown below:

```
service-template DOT1X
 access-group SVC1_ACL
 redirect url www.cisco.com match URL_REDIRECT_ACL
 inactivity-timer 60
 absolute-timer 300
!
ip access-list extended URL_REDIRECT_ACL
 permit tcp any host 5.5.5.5 eq www
```

Control Policy Configuration

The following example shows a control policy that activates the service template named DOT1X with replace-all enabled. The successfully activated template will replace the existing authorization data and any service template previously applied to the session.

```
policy-map type control subscriber POSTURE_VALIDATION
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using dot1x priority 10
   20 authenticate using webauth priority 20
 event authentication-success match-all
  10 class DOT1X do-all
   10 terminate webauth
   20 activate service-template DOT1X replace-all
```

Example: Activating a Service Template for Fallback Service

Local Service Template Configuration

The following example shows the configuration of a service template defined locally on the device. This template contains attributes that are applied to sessions that use the control policy named POSTURE_VALIDATION, shown below:

```
service-template FALLBACK
 description fallback service
 access-group ACL_2
 redirect url www.cisco.com
 inactivity-timer 15
 absolute-timer 15
 tag TAG_2
```

Control Policy Configuration

The following example shows a control policy that runs authentication methods dot1x and MAB. If dot1x authentication fails, MAB authentication is attempted. If MAB fails, the system provides a default authorization profile using the FALLBACK template.

```
policy-map type control subscriber POSTURE_VALIDATION
 event session-started match-all
  10 class always do-all
   10 authenticate using dot1x
 event authentication-failure match-all
  10 class DOT1X do-all
   10 authenticate using mab
  20 class MAB do-all
   10 activate service-template FALLBACK
```

Example: Deactivating a Service Template

Access Control List Configuration

The following example shows the configuration of an access control list (ACL) that is used by the local service template named LOW_IMPACT_TEMPLATE, shown below.

```
ip access-list extended LOW_IMPACT_ACL
 permit udp any any eq bootps
 permit tcp any any eq www
 permit tcp any any eq 443
 permit ip any 172.30.0.0 0.0.255.255
```

Local Service Template Configuration

The following example shows the configuration of the local service template that provides limited access to all hosts even when authentication fails.

```
service-template LOW_IMPACT_TEMPLATE
 description Service template for Low impact mode
 access-group LOW_IMPACT_ACL
 inactivity-timer 60
 tag LOW_IMPACT_TEMPLATE
```

Control Policy Configuration

The following example shows the configuration of a control policy that uses the template named LOW_IMPACT_TEMPLATE to provide limited access to all hosts even when authentication fails. If authentication succeeds, the policy manager removes the service template and provides access based on the policies downloaded by the RADIUS server.

```
class-map type control subscriber match-all DOT1X_MAB_FAILED
 no-match result-type method dot1x success
 no-match result-type method mab success
!
policy-map type control subscriber CONCURRENT_DOT1X_MAB_LOW_IMP_MODE
 event session-started match-all
  10 class always do-until-failure
   10 authorize
   20 activate service-template LOW_IMPACT_TEMPLATE
   30 authenticate using mab
   40 authenticate using dot1x
 event authentication-success match-all
  10 class always do-until-failure
   10 deactivate service-template LOW_IMPACT_TEMPLATE
 event authentication-failure match-first
  10 class DOT1X_MAB_FAILED do-until-failure
   10 authorize
   20 terminate dot1x
   30 terminate mab
 event agent-found match-all
  10 class always do-until-failure
   10 authenticate using dot1x
 event inactivity-timeout match-all
  10 class always do-until-failure
   10 clear-session
```

CHAPTER 107

Interface Templates

Interface Templates

An interface template provides a mechanism to configure multiple commands at the same time and associate it with a target such as an interface. An interface template is a container of configurations or policies that can be applied to specific ports.

Restrictions for Interface Templates

- Interface templates are not applicable for wireless sessions.
- Remote storing and downloading of templates is not supported.
- The same configuration cannot be used for port and interface template on the switch.

Information About Interface Templates

About Interface Templates

An interface template is a container of configurations or policies that can be applied to specific ports. When an interface template is applied to an access port, it impacts all traffic that is exchanged on the port.

There are two types of interface templates; user and builtin templates. Builtin templates are created by the system.

You can modify builtin templates. If you delete a modified builtin template the system restores the original definition of the template.

The following are the available builtin templates:

- AP_INTERFACE_TEMPLATE (Access Point)
- DMP_INTERFACE_TEMPLATE (Digital Media Player)
- IP_CAMERA_INTERFACE_TEMPLATE
- IP_PHONE_INTERFACE_TEMPLATE

- LAP_INTERFACE_TEMPLATE (Lightweight Access Point)
- MSP_CAMERA_INTERFACE_TEMPLATE
- MSP_VC_INTERFACE_TEMPLATE (Video Conferencing)
- PRINTER_INTERFACE_TEMPLATE
- ROUTER_INTERFACE_TEMPLATE
- SWITCH_INTERFACE_TEMPLATE
- TP_INTERFACE_TEMPLATE (TelePresence)

Following is an example of a builtin interface template:

```
Template Name       : IP_CAMERA_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
```

You can also create specific user templates with the commands that you want to include.

Note The template name must not contain spaces.

You can create an interface template using the **template** command in global configuration mode. In template configuration mode, enter the required commands. The following commands can be entered in template configuration mode:

Command	Description
access-session	Configures access session specific interface commands.
authentication	Configures authentication manager Interface Configuration commands.
carrier-delay	Configures delay for interface transitions.
dampening	Enables event dampening.
default	Sets a command to its defaults.
description	Configures interface-specific description.
dot1x	Configures interface configuration commands for IEEE 802.1X.
hold-queue	Sets hold queue depth.
ip	Configures IP template.

Command	Description
keepalive	Enables keepalive.
load-interval	Specifies interval for load calculation for an interface.
mab	Configures MAC authentication bypass Interface.
mls	Enables multilayer switching configurations.
peer	Configures peer parameters for point to point interfaces.
priority-queue	To set the priority-queue size for a template.
queue-set	Configures the QoS queue set on a template.
radius-server	Enables RADIUS server configurations.
service-policy	Configures CPL service policy.
source	Gets configurations from another source.
spanning-tree	Configures spanning tree subsystem
storm-control	Configures storm control.
subscriber	Configures subscriber inactivity timeout value.
switchport	Sets switching mode configurations
trust	Sets trust value for the interface.

Note

- System builtin templates are not displayed in the running configuration. These templates show up in the running configuration only if you edit them.
- The stateful switchover fails if **access-session** and **swithcport mode access** are both configured in an interface template. To avoid the switchover failure, configure the **switchport mode access** command on the interface, instead of in an interface template.
- When you configure an interface template, it is recommended that you enter all the required dependent commands on the same template. It is not recommended to configure the dependent commands on two different templates.

Binding an Interface Template to a Target

Each template can be bound to a target. Template binding or sourcing can be either static or dynamic. Static binding of a template involves binding the template to a target, like an interface. Only one template can be bound at a time using static binding. Static binding of another template to the same target will unbind the previously bound template. To configure static binding, use the **source template** command in interface configuration mode.

Any number of templates can be bound dynamically to a target. To configure dynamic binding using builtin policy maps and parameter maps, enable the autoconf feature using the **autoconf enable** command.

Note You can have statically and dynamically bind templates on the same interface at a time.

Priority for Configurations Using Interface Templates

Configuration applied through dynamically-bound templates has the highest priority, followed by configuration applied directly on the interface, and then configuration applied through statically-bound templates. When similar commands are present at different priority levels, the one at the highest priority is applied. If a configuration at a higher priority level is not applied, then the configuration with the next highest priority is applied to the target.

Multiple templates can be dynamically bound to a target. When multiple templates are dynamically bound, the template that is applied last has the highest priority.

To delete a template, you must remove the binding to all targets. If you bind a template that does not exist, a new template is created with no configurations.

How to Configure Interface Templates

Configuring Interface Templates

Perform the following task to create user interface templates:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**template** *name* **Example:** `Device(config)# template user-template1`	Creates a user template and enters template configuration mode. **Note** Builtin template are system-generated.
Step 4	**load-interval** *interval* **Example:**	Configures the sampling interval for statistics collections on the template. **Note**

	Command or Action	Purpose
	`Device(config-template)# load-interval 60`	Builtin template are system-generated.
Step 5	**description** *description* **Example:** `Device(config-template)# description This is a user template`	Configures the description for the template.
Step 6	**keepalive** *number* **Example:** `Device(config-template)# Keepalive 60`	Configures the keepalive timer.
Step 7	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Static Binding for Interface Templates

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies the interface type and number and enters interface configuration mode.
Step 4	**source template** *name* **Example:** `Device(config-if)# source template user-template1`	Statically applies an interface template to a target.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example

To verify static binding use the **show running-config interface** *int-name* and the **show derived-config interface** *int-name* commands.

```
Device# show running-config interface GigabitEthernet 1/1

Building configuration...

Current configuration : 71 bytes
!
interface GigabitEthernet1/1
source template user-template1
end

Device# show derived-config interface GigabitEthernet 1/1
Building configuration...

Derived configuration : 108 bytes
!
interface GigabitEthernet1/1
description This is a user template
load-interval 60
keepalive 60
end
```

Configuring Dynamic Binding of Interface Templates

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:**	Specifies the interface type and number and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface GigabitEthernet 1/1`	
Step 4	**service-policy type control subscriber** *policymap-name* **Example:** `Device(config-if)# service-policy type control subscriber POLICY-Gi1/1`	Dynamically applies an interface template to a target.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying an Interface Template

Use one or more of the commands listed below to verify the interface template configuration.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **show template interface all** {**all** | **binding** {*temp-name* | **all** | **target** *int-name*} | **brief** }
Shows all interface template configurations.

Step 3 **show template interface source** {**built-in** [**original**] | **user**} {*temp-name* | **all**}}
Shows interface template source configurations.

Step 4 **show template service**{**all** | **binding target** *int-name* | **brief** | **source** {**aaa** | **built-in** | **user** {**temp-name** | **all**}}
Shows all interface template service configurations.

Configuration Examples for Interface Templates

Example: Configuring User Interface Templates

Example: Configuring User Templates

```
Device# enable
Device (config)# configure terminal
Device(config)# template user-template1
Device(config-template)# load-interval 60
Device(config-template)# description This is a user template
Device(config-template)# Keepalive 60
Device(config)# end
```

Example: Sourcing Interface Templates

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 1/1
Device(config-if)# source template user-template1
Device(config-if)# end
```

Example: Dynamically Binding Interface Templates

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 1/1
Device(config-if)# service-policy type control subscriber POLICY_Gi1/1
Device(config-if)# end
```

CHAPTER 108

Autoconf

Autoconf

Autoconf is a solution that can be used to manage port configurations for data or voice VLAN, quality of service (QoS) parameters, storm control, and MAC-based port security on end devices that are deployed in the access layer of a network.

Prerequisites for Autoconf

- Before enabling Autoconf, disable the Auto SmartPort (ASP) macro, device classifier, and then access the session monitor.

Restrictions for Autoconf

- Interface templates are not applicable for wireless sessions.
- When the Autoconf feature is enabled using the **autoconf enable** command, the default Autoconf service policy is applied to all interfaces. No other service policy can be applied globally using the **service-policy** command. To apply a different service policy, you must disable Autoconf on that interface. When a service policy is applied globally, you must disable it before enabling the Autoconf feature.
- When both local (interface-level) and global service policies exist, the local policy take precedence. Events in the local service policy are handled and the global service policy is not applied. The global service policy comes into effect only when the local policy is removed.
- Service templates cannot be applied to interfaces, and interface templates cannot be applied to service instances.
- Only one service template can be nested inside an interface template.

Information About Autoconf

Benefits of Autoconf

The Autoconf feature permits hardbinding between the end device and the interface. Autoconf falls under the umbrella of the Smart Operations solution. Smart Operations is a comprehensive set of capabilities that can simplify and improve LAN switch deployment. Smart Operations help organizations deliver operational excellence and scale services on the network.

The Autoconf feature automatically applies the needed configurations on the device ports to enable the efficient performance of each directly connected end device using a set of interface configurations that are configured inside an interface template.

- Autoconf efficiently applies commands to an interface because the parser does not need to parse each command each time.
- Configurations that are applied through the Autoconf feature can be reliably removed from a port without impacting previous or subsequent configurations on the port.
- The Autoconf feature provides built-in and user-defined configurations using interface and service templates. Configurations applied through templates can be centrally updated with a single operation.
- Using the Autoconf feature, a configuration can be applied to ports and access sessions.
- The Autoconf feature reduces ongoing maintenance for devices and attached end devices by making them intuitive and autoconfigurable. This reduces operation expenses (OPEX) and lowers the total cost of ownership (TCO).

Identity Session Management and Templates

A key advantage of the Autoconf feature is that the core session management capability is decoupled from the application-specific logic; thus, allowing the same framework to be used regardless of the criteria for policy determination or the nature of the policies applied.

The identity session management infrastructure allows configurations and/or policies to be applied as templates.

Both service and interface templates are named containers of configuration and policy. Service templates may be applied only to access sessions, while interface templates may be applied only to ports. When a service template is applied to an access session, the contained configuration/policy is applied only to the target session and has no impact on other sessions that may be hosted on the same access port. Similarly, when an interface template is applied to an access port, it impacts all traffic exchanged on the port.

The Autoconf feature uses a set of built-in maps and built-in templates. The built-in templates are designed based on best practices for interface configurations. Built-in templates can be modified by the user to include customized configurations, limiting the need to create a new template.

The templates created by users are referred to as user-defined templates. User-defined templates can be defined on the device and can be mapped to any built-in or user-defined trigger.

Use the **show derived-config** command, to view the overall applied configurations applied by Autoconf template and manual configuration. The interface commands shown in the output of **show running-config interface** *type number* command are not necessarily the operational configuration. The Autoconf feature dynamically applies a template to the interface, and overrides any conflicting static configuration that is already applied.

Autoconf Operation

Autoconf uses the Device Classifier to identify the end devices that are connected to a port.

The Autoconf feature uses the device classification information gleaned from Cisco Discovery Protocol, LLDP, DHCP, MAC addresses, and the Organizationally Unique Identifier (OUI) that is identified by the Device Classifier.

The Device Classifier provides improved device classification capabilities and accuracy, and increased device visibility for enhanced configuration management.

Device classification is enabled when you enable the Autoconf feature using **autoconf enable** command in global configuration mode.

The device detection acts as an event trigger, which in turn applies the appropriate automatic template to the interface.

The Autoconf feature is based on a three-tier hierarchy.

- A policy map identifies the trigger type for applying the Autoconf feature.
- A parameter map identifies the appropriate template that must be applied, based on the end device.
- The templates contain the configurations to be applied.

The Autoconf built-in templates and triggers perform the these three steps automatically.

The Autoconf feature provides the following built-in templates:

- AP_INTERFACE_TEMPLATE
- DMP_INTERFACE_TEMPLATE
- IP_CAMERA_INTERFACE_TEMPLATE
- IP_PHONE_INTERFACE_TEMPLATE
- LAP_INTERFACE_TEMPLATE
- MSP_CAMERA_INTERFACE_TEMPLATE
- MSP_VC_INTERFACE_TEMPLATE
- PRINTER_INTERFACE_TEMPLATE
- ROUTER_INTERFACE_TEMPLATE
- SWITCH_INTERFACE_TEMPLATE
- TP_INTERFACE_TEMPLATE

Note By default built-in templates are not displayed under running configuration. The built-in templates show in the running configuration only if you edit them.

The template that is selected is based on parameter map information applied to an interface. This information can be based on the following criteria:

- End Device type

- MAC address
- OUI
- User role
- Username

The Autoconf feature provides one built-in parameter map BUILTIN_DEVICE_TO_TEMPLATE with the following configuration:

```
Parameter-map name: BUILTIN_DEVICE_TO_TEMPLATE
 Map: 10 map device-type regex "Cisco-IP-Phone"
  Action(s):
   20 interface-template IP_PHONE_INTERFACE_TEMPLATE
 Map: 20 map device-type regex "Cisco-IP-Camera"
  Action(s):
   20 interface-template IP_CAMERA_INTERFACE_TEMPLATE
 Map: 30 map device-type regex "Cisco-DMP"
  Action(s):
   20 interface-template DMP_INTERFACE_TEMPLATE
 Map: 40 map oui eq "00.0f.44"
  Action(s):
   20 interface-template DMP_INTERFACE_TEMPLATE
 Map: 50 map oui eq "00.23.ac"
  Action(s):
   20 interface-template DMP_INTERFACE_TEMPLATE
 Map: 60 map device-type regex "Cisco-AIR-AP"
  Action(s):
   20 interface-template AP_INTERFACE_TEMPLATE
 Map: 70 map device-type regex "Cisco-AIR-LAP"
  Action(s):
   20 interface-template LAP_INTERFACE_TEMPLATE
 Map: 80 map device-type regex "Cisco-TelePresence"
  Action(s):
   20 interface-template TP_INTERFACE_TEMPLATE
 Map: 90 map device-type regex "Surveillance-Camera"
  Action(s):
   10 interface-template MSP_CAMERA_INTERFACE_TEMPLATE
 Map: 100 map device-type regex "Video-Conference"
  Action(s):
   10 interface-template MSP_VC_INTERFACE_TEMPLATE
```

Note Use the **show parameter-map type subscriber attribute-to-service All** command to view the configuration for the built-in parameter map.

The Autoconf feature provides one built-in policy map BUILTIN_AUTOCONF_POLICY with the following configuration:

```
BUILTIN_AUTOCONF_POLICY
  event identity-update match-all
    10 class always do-until-failure
      10 map attribute-to-service table BUILTIN_DEVICE_TO_TEMPLATE
```

Note Use the **show policy-map type control subscriber BUILTIN_AUTOCONF_POLICY** command to view the configuration for the built-in policy map.

You can also manually create policy maps, parameter maps, and templates.

When a trigger is created that is based on specific user information, a local 802.1X Cisco Identity Services Engine (ISE) server authenticates it ensuring the security of the operation.

An interface template can be dynamically activated (on an interface) using any of the following methods:

- RADIUS CoA—While Change of Authorization (CoA) commands are targeted to one or more access sessions, any referenced template must be applied to the interface hosting the referenced session.
- RADIUS Access-Accept for client authentication or authorization—Any referenced interface template returned in an Access-Accept must be applied to the port that is hosting the authorized access session.
- Service template—If an interface template is referenced in a service template that is either locally defined or sourced from the AAA server, the interface template must be applied to the interface hosting any access-session on which the service template is applied (add a new command for interface template reference from within a locally defined service template).
- Subscriber control-policy action—A mapping action under the subscriber control policy activates service and/or interface template (as referenced in a parameter map) based on the type of filter, and removes any templates associated with a previous policy.
- Device-to-template parameter map—A subscriber parameter map that allows the filter type to service and/or interface template mappings to be specified in an efficient and readable manner.

Advantages of Using Templates

Using templates for autoconfiguration has the following benefits:

- Templates are parsed once when they are being defined. This makes dynamic application of the templates very efficient.
- Templates can be applied to an Ethernet interface that is connected to an end device, based on the type of the end device.
- Service templates allow the activation of session-oriented features, whereas interface templates apply configurations to the interface that is hosting a session.
- Service templates are applied to access sessions and hence only impact the traffic exchanged with a single endpoint on a port.
- Startup and running configurations of the device are not modified by the dynamic application of the template.
- Policy application is synchronized with the access-session life cycle, which is tracked by the framework by using all available techniques, including just link-up/link-down.
- Templates can be updated with a single operation. All applied instances of the templates are updated.
- Constituent commands of the templates do not appear in the running configuration.
- Templates can be removed with no impact on previous or subsequent configurations.
- Template application is acknowledged, allowing for synchronization and performing remedial actions where failures occur.
- Data VLAN, quality of service (QoS) parameters, storm control, and MAC-based port security are configured automatically based on the end device that is connected to the switch.

- The switch port is cleaned up completely by removing configurations when the device is disconnected from a port.
- Human error is reduced in the installation and configuration process.

Autoconf Functionality

The Autoconf feature is disabled by default in global configuration mode. When you enable the Autoconf feature in global configuration mode, it is enabled by default at the interface level. The built-in template configurations are applied based on the end devices detected on all interfaces.

Use the **access-session inherit disable autoconf** command to manually disable Autoconf at the interface level, even when Autoconf is enabled at the global level.

If you disable Autoconf at the global level, all interface-level configurations are disabled.

Global	Interface Level	AutoConf Status
Disable	Disable	No automatic configurations are applied when an end device is connected.
Enable	Enabled by default	If Autoconf is enabled at the global level, it is enabled at the interface level by default. Built-in template configurations are applied based on the end devices that are detected on all interfaces.
Enable	Disable	Enabled at global level. Disabled at interface level. No automatic configurations are applied when an end device is connected to the interface on which Autoconf is disabled.

Autoconf allows you to retain the template even when the link to the end device is down or the end device is disconnected, by configuring the Autoconf sticky feature. Use the **access-session interface-template sticky** command to configure the Autoconf sticky feature in global configuration mode. The Autoconf sticky feature avoids the need for detecting the end device and applying the template every time the link flaps or device is removed and connected back.

The **access-session interface-template sticky** command is mandatory to apply an inbuilt template that contains **access-session** commands on an interface. Configure the **access-session interface-template sticky** command to apply interface template on a port using a service policy.

If you want to disable the Autoconf feature on a specific interface, use the **access-session inherit disable interface-template-sticky** command in interface configuration mode.

How to Configure Autoconf

Applying a Built-in Template to an End Device

The following task shows how to apply a built-in template on an interface that is connected to an end device, for example, a Cisco IP phone.

Before you begin

Make sure that the end device, for example, a Cisco IP phone, is connected to the switch port.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)#configure terminal`	Enters global configuration mode.
Step 3	**autoconf enable** **Example:** `Device(config)#autoconf enable`	Enables the Autoconf feature.
Step 4	**end** **Example:** `Device(config)#end`	Exits global configuration mode and enters privileged EXEC mode.
Step 5	(Optional) **show device classifier attached interface** *interface-type interface-number* **Example:** `Device#show device classifier attached interface Gi1/1`	Displays whether the end device is classified by the device classifier with correct attributes.
Step 6	**show template binding target** *interface-type interface-number* **Example:** `Device#show template binding target gi1/1`	Displays the configuration applied through the template on the interface.

Applying a Modified Built-in Template to an End Device

The following task shows how to modify a built-in template when multiple wireless access points and IP cameras are connected to a switch.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device(config)# configure terminal`	
Step 3	**template** *template-name* **Example:** `Device(config)# template AP_INTERFACE_TEMPLATE`	Enters template configuration mode for the builtin template.
Step 4	**switchport access vlan** *vlan-id* **Example:** `Device(config-template)# switchport access vlan 20`	Sets the VLAN when the interface is in access mode.
Step 5	**description** *description* **Example:** `Device(config-template)# description modifiedAP`	Modifies the description of the built-in template.
Step 6	**exit** **Example:** `Device(config-template)# exit`	Exits template configuration mode and enters global configuration mode.
Step 7	**autoconf enable** **Example:** `Device(config)# autoconf enable`	Enables the Autoconf feature.
Step 8	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.
Step 9	**show template interface binding all** **Example:** `Device# show template interface binding all`	Displays whether the template is applied on the interface.

Migrating from ASP to Autoconf

Before you begin

Verify that the AutoSmart Port (ASP) macro is running using the **show running-config | include macro auto global** command.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no macro auto global processing** **Example:** `Device(config)# no macro auto global processing`	Disables ASP on a global level.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**clear macro auto configuration all** **Example:** `Device# clear macro auto configuration all`	Clears macro configurations for all interfaces.
Step 6	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 7	**autoconf enable** **Example:** `Device(config)# autoconf enable`	Enables the Autoconf feature.
Step 8	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuration Examples for Autoconf

Example: Applying a Built-in Template to an End Device

The following example shows how to apply a built-in template to an end device connected to an interface.

```
Device> enable
Device(config)# configure terminal
Device(config)# autoconf enable
Device(config)# end
Device# show device classifier attached interface Gi1/1
Device# show template binding target GigabitEthernet 1/1
```

Example: Applying a Modified Built-in Template to an End Device

The following example shows how to modified built-in template and verify the configuration:

```
Device> enable
Device(config)# configure terminal
Device(config)# template AP_INTERFACE_TEMPLATE
Device(config-template)# switchport access vlan 20
Device(config-template)# description modifiedAP
Device(config-template)# exit
Device(config)# autoconf enable
Device(config)# end
Device# show template interface binding all
```

Example: Migrating from ASP Macros to Autoconf

The following example shows how to migrate from ASP to Autoconf:

```
Device> enable
Device# configure terminal
Device(config)# no macro auto global processing
Device(config)# exit
Device# clear macro auto configuration all
Device# configure terminal
Device(config)# autoconf enable
Device(config)# end
```

CHAPTER 109

Critical Voice VLAN Support

Critical Voice VLAN Support

The Critical Voice VLAN Support feature directs phone traffic to the configured voice VLAN of a port if the authentication server becomes unreachable.

With normal network connectivity, when an IP phone successfully authenticates on a port, the authentication server directs the phone traffic to the voice domain of the port. If the authentication server becomes unreachable, IP phones cannot authenticate the phone traffic. In multidomain authentication (MDA) mode or multiauthentication mode, you can configure the Critical Voice VLAN Support feature to direct phone traffic to the configured voice VLAN of the port. The phone is authorized as an unknown domain. Both data and voice are enabled for the phone.

Restrictions for Critical Voice VLAN Support

- Different VLANs must be configured for voice and data.
- The voice VLAN must be configured on a device.
- The Critical Voice VLAN Support feature does not support standard Access Control Lists (ACLs) on the switch port.

Information About Critical Voice VLAN Support

Critical Voice VLAN Support in Multidomain Authentication Mode

If a critical voice VLAN is deployed using an interface in multidomain authentication (MDA) mode, the host mode is changed to multihost and the first phone device is installed as a static forwarding entries. Any additional phone devices are installed as dynamic forwarding entry in the Host Access Table (HAT).

Note

- If a critical port is already authorized and reauthentication occurs, the switch puts the port in the critical-authentication state in the current VLAN, which might be the one previously assigned by the RADIUS server.
- Inaccessible authentication bypass is compatible with guest VLAN. When a guest VLAN is enabled on a 802.1X port, the features interact as follows: if all RADIUS servers are not available and if a client is connected to a critical port and was previously assigned to a guest VLAN, the switch keeps the port in the guest VLAN.

Critical Voice VLAN Support in Multiauthentication Mode

If the critical authentication feature is deployed in multiauthentication mode, only one phone device will be allowed and a second phone trying to authorize will trigger a violation.

The **show access-session** command displays the critical voice client data. A critically authorized voice client in multiauthentication host mode will be in the “authz success” and “authz fail” state.

Note If critical voice is required, then critical data should be configured too. Otherwise, the critical voice client will be displayed in the “authz fail” state while the voice VLAN will be open.

Critical Voice VLAN Support in a Service Template

On enterprise Edge (eEdge) devices, the critical access of phones is configured by activating a critical service template when the authentication server becomes unreachable. The voice feature plug-in registers with the Enterprise Policy Manager (EPM) by using an authentication, authorization, and accounting (AAA) voice attribute, and it allows unconditional access to the voice VLAN while the AAA services are unavailable.

To enable critical voice VLAN support, the critical authentication of phones must be configured using a combination of control policy rules and a service template.

When the authentication server is unavailable and the host is unauthorized, the AAA attribute device-traffic-type is not populated. The phone is authorized as an unknown domain, and both the data and voice VLAN are enabled for this device, allowing the device to handle voice traffic.

How to Configure Critical Voice VLAN Support

Configuring a Critical Voice VLAN in a Service Template

Perform this task on a port to configure critical voice VLAN support using a service template.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service-template** *template-name* **Example:** `Device(config)# service-template CRITICAL-DATA`	Defines a template that contains a set of service policy attributes to apply to subscriber sessions and enters service template configuration mode.
Step 4	**vlan** *vlan-id* **Example:** `Device(config-service-template)# vlan 116`	Assigns a VLAN to a subscriber session.
Step 5	**exit** **Example:** `Device(config-service-template)# exit`	Exits service template configuration mode and returns to global configuration mode.
Step 6	**service-template** *template-name* **Example:** `Device(config)# service-template CRITICAL-VOICE`	Defines a template that contains a set of service policy attributes to apply to subscriber sessions and enters service template configuration mode.
Step 7	**voice vlan** **Example:** `Device(config-service-template)# voice vlan`	Assigns a critical voice VLAN to a subscriber session.
Step 8	**exit** **Example:** `Device(config-service-template)# exit`	Exits service template configuration mode and returns to global configuration mode.
Step 9	**class-map type control subscriber** {**match-all** | **match-any** | **match-none**} *control-class-name* **Example:** `Device(config)# class-map type control subscriber match-all AAA-SVR-DOWN-UNAUTHD-HOST`	Creates a control class, which defines the conditions under which the actions of a control policy are executed and enters control class-map filter configuration mode.
Step 10	**match result-type** [**method** {**dot1x** | **mab** | **webauth**}] *result-type* **Example:**	Creates a condition that returns true based on the specified authentication result.

	Command or Action	Purpose
	`Device(config-filter-control-classmap)# match result-type aaa-timeout`	
Step 11	**match authorization-status** {**authorized** | **unauthorized**} **Example:** `Device(config-filter-control-classmap)# match authorization-status unauthorized`	Creates a condition that returns true based on the authorization status of a session.
Step 12	**exit** **Example:** `Device(config-filter-control-classmap)# exit`	Exits control class-map filter configuration mode and returns to global configuration mode.
Step 13	**class-map type control subscriber** {**match-all** | **match-any** | **match-none**} *control-class-name* **Example:** `Device(config)# class-map type control subscriber match-all AAA-SVR-DOWN-AUTHD-HOST`	Creates a control class, which defines the conditions under which the actions of a control policy are executed and enters control class-map filter configuration mode.
Step 14	**match result-type** [**method** {**dot1x** | **mab** | **webauth**}] *result-type* **Example:** `Device(config-filter-control-classmap)# match result-type aaa-timeout`	Creates a condition that returns true based on the specified authentication result.
Step 15	**match authorization-status** {**authorized** | **unauthorized**} **Example:** `Device(config-filter-control-classmap)# match authorization-status authorized`	Creates a condition that returns true based on the authorization status of a session.
Step 16	**end** **Example:** `Device(config-filter-control-classmap)# end`	Exits control class-map filter configuration mode and returns to privileged EXEC mode.

Activating Critical Voice VLAN

Perform the following task to activate a critical voice VLAN that is configured on a service template.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type control subscriber** *control-policy-name* **Example:** `Device(config)# policy-map type control subscriber cisco-subscriber`	Defines a control policy for subscriber sessions and enters control policy-map event configuration mode.
Step 4	**event authentication-failure** [**match-all** | **match-first**] **Example:** `Device(config-event-control-policymap)# event authentication-failure match-first`	Specifies the type of event that triggers actions in a control policy if all authentication events are a match and enters control policy-map class configuration mode.
Step 5	*priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**] **Example:** `Device(config-class-control-policymap)# 10 class AAA-SVR-DOWN-UNAUTHD-HOST do-until-failure`	Specifies that the control class should execute the actions in a control policy, in the specified order, until one of the actions fails, and enters control policy-map action configuration mode.
Step 6	*action-number* **activate** {**policy type control subscriber** *control-policy-name* | **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** [**replace-all**]]} **Example:** `Device(config-action-control-policymap)# 10 activate service-template CRITICAL-DATA`	Activates a control policy associated with the VLAN on a subscriber session.
Step 7	*action-number* **activate** {**policy type control subscriber** *control-policy-name* | **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** [**replace-all**]]} **Example:** `Device(config-action-control-policymap)# 20 activate service-template CRITICAL-VOICE`	Activates a control policy associated with the voice VLAN on a subscriber session.

	Command or Action	Purpose
Step 8	*action-number* **authorize** **Example:** `Device(config-action-control-policymap)# 30 authorize`	Initiates the authorization of a subscriber session.
Step 9	*action-number* **pause reauthentication** **Example:** `Device(config-action-control-policymap)# 40 pause reauthentication`	Pauses the reauthentication process after an authentication failure.
Step 10	**exit** **Example:** `Device(config-action-control-policymap)# exit`	Exits control policy-map action configuration mode and enters control policy-map class configuration mode.
Step 11	*priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**] **Example:** `Device(config-class-control-policymap)# 20 class AAA-SVR-DOWN-AUTHD-HOST`	Specifies that the control class should execute the actions in a control policy, in the specified order, until one of the actions fails, and enters control policy-map action configuration mode.
Step 12	*action-number* **pause reauthentication** **Example:** `Device(config-action-control-policymap)# 10 pause reauthentication`	Pauses the reauthentication process after an authentication failure.
Step 13	**end** **Example:** `Device(config-action-control-policymap)# exit`	Exits control policy-map action configuration mode and enters privileged EXEC mode.

Configuration Examples for Critical Voice VLAN Support

Example: Configuring a Voice VLAN in a Service Template

```
Device> enable
Device# configure terminal
Device(config)# service-template CRITICAL-DATA
Device(config-service-template)# vlan 116
Device(config-service-template)# exit
Device(config)# service-template CRITICAL-VOICE
Device(config-service-template)# voice vlan
Device(config-service-template)# exit
Device(config)# class-map type control subscriber match-all AAA-SVR-DOWN-UNAUTHD-HOST
Device(config-filter-control-classmap)# match result-type aaa-timeout
Device(config-filter-control-classmap)# match authorization-status unauthorized
Device(config-filter-control-classmap)# exit
```

```
Device(config)# class-map type control subscriber match-all AAA-SVR-DOWN-AUTHD-HOST
Device(config-filter-control-classmap)# match result-type aaa-timeout
Device(config-filter-control-classmap)# match authorization-status authorized
Device(config-filter-control-classmap)# end
```

Example: Activating a Critical Voice VLAN on a Service Template

```
Device> enable
Device# configure terminal
Device(config)# policy-map type control subscriber cisco-subscriber
Device(config-event-control-policymap)# event authentication-failure match-first
Device(config-class-control-policymap)# 10 class AAA-SVR-DOWN-UNAUTHD-HOST do-until-failure
Device(config-action-control-policymap)# 10 activate service-template CRITICAL-DATA
Device(config-action-control-policymap)# 10 activate service-template CRITICAL-VOICE
Device(config-action-control-policymap)# 30 authorize
Device(config-action-control-policymap)# 40 pause reauthentication
Device(config-action-control-policymap)# exit
Device(config-class-control-policymap)# 20 class AAA-SVR-DOWN-AUTHD-HOST
Device(config-action-control-policymap)# 10 pause reauthentication
Device(config-action-control-policymap)# end
```

CHAPTER 110

Configuring Local Authentication Using LDAP

Configuring Local Authentication Using LDAP

This module provides information about configuring local authentication for Cisco Identity Based Networking Services.

Information About Local Authentication Using LDAP

Local Authentication Using LDAP

Local authentication using Lightweight Directory Access Protocol (LDAP) allows an endpoint to be authenticated using 802.1X, MAC authentication bypass (MAB), or web authentication with LDAP as a backend. Local authentication in Identity-Based Networking Services also supports associating an authentication, authorization, and accounting (AAA) attribute list with the local username for wireless sessions.

How to Configure Local Authentication Using LDAP

Configuring Local Authentication Using LDAP

Perform this task to specify the AAA method list for local authentication and to associate an attribute list with a local username.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables the authentication, authorization, and accounting (AAA) access control model.
Step 4	**aaa local authentication** {*method-list-name* | **default**} **authorization** {*method-list-name* | **default**} **Example:** `Device(config)# aaa local authentication default authorization default`	Specifies the method lists to use for local authentication and authorization from a LDAP server.
Step 5	**username** *name* **aaa attribute list** *aaa-attribute-list* [**password** *password*] **Example:** `Device(config)# username USER_1 aaa attribute list LOCAL_LIST password CISCO`	Associates a AAA attribute list with a local username.
Step 6	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring MAC Filtering Support

Perform this task to set the RADIUS compatibility mode, the MAC delimiter, and the MAC address as the username to support MAC filtering.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables the authentication, authorization, and accounting (AAA) access control model.
Step 4	**aaa group server radius** *group-name* **Example:**	Groups different RADIUS server hosts into distinct lists.

	Command or Action	Purpose
	`Device(config)# aaa group server radius RAD_GROUP1`	
Step 5	**subscriber mac-filtering security-mode** {**mac** | **none** | **shared-secret**} **Example:** `Device(config-sg-radius)# subscriber mac-filtering security-mode mac`	Specifies the RADIUS compatibility mode for MAC filtering. • The default value is **none**.
Step 6	**mac-delimiter** {**colon** | **hyphen** | **none** | **single-hyphen**} **Example:** `Device(config-sg-radius)# mac-delimiter hyphen`	Specifies the MAC delimiter for RADIUS compatibility mode. • The default value is **none**.
Step 7	**exit** **Example:** `Device(config-sg-radius)# exit`	Exits server group configuration mode and returns to global configuration mode.
Step 8	**username** *mac-address* **mac** [**aaa attribute list** *aaa-attribute-list*] **Example:** `Device(config)# username 00-22-WP-EC-23-3C mac aaa attribute list AAA_list1`	Allows a MAC address to be used as the username for MAC filtering done locally.
Step 9	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Configuration Examples for Local Authentication Using LDAP

Example: Configuring Local Authentication Using LDAP

The following example shows a configuration for local authentication:

```
!
username USER_1 password 0 CISCO
username USER_1 aaa attribute list LOCAL_LIST
aaa new-model
aaa local authentication EAP_LIST authorization EAP_LIST
!
```

Example: Configuring MAC Filtering Support

The following example shows a configuration for MAC filtering:

```
username 00-22-WP-EC-23-3C mac aaa attribute list AAA_list1
!
aaa new-model
aaa group server radius RAD_GROUP1
subscriber mac-filtering security-mode mac
mac-delimiter hyphen
```

CHAPTER 111

Web Authentication Redirection to Original URL

Web Authentication Redirection to Original URL Overview

The Web Authentication Redirection to Original URL feature enables networks to redirect guest users to the URL that they had originally requested. This feature is enabled by default and requires no configuration.

Guest networks are network connections provided by an enterprise to allow their guests to gain access to the Internet and to their own enterprise networks without compromising the security of the host enterprise. Guest users of an enterprise network can connect to the guest access network through either a wired Ethernet connection or a wireless connection.

Guest access uses a captive portal to gather all web requests made by guests and redirect these requests to one of the guest on-boarding web pages. When guests successfully complete the guest workflow, they are redirected to the page that they had originally requested.

The originally requested URL is passed as metadata along with the Cisco Identity Services Engine (ISE) guest access redirect URL. The Cisco ISE is a security policy management and control platform. It automates and simplifies access control and security compliance for wired, wireless, and VPN connectivity. The requested URL is added at the end of the Cisco ISE guest URL so that the device can send the redirect URL to the guest client. The Cisco ISE parses the URL and redirects the guest to the original URL after completing the on-boarding.

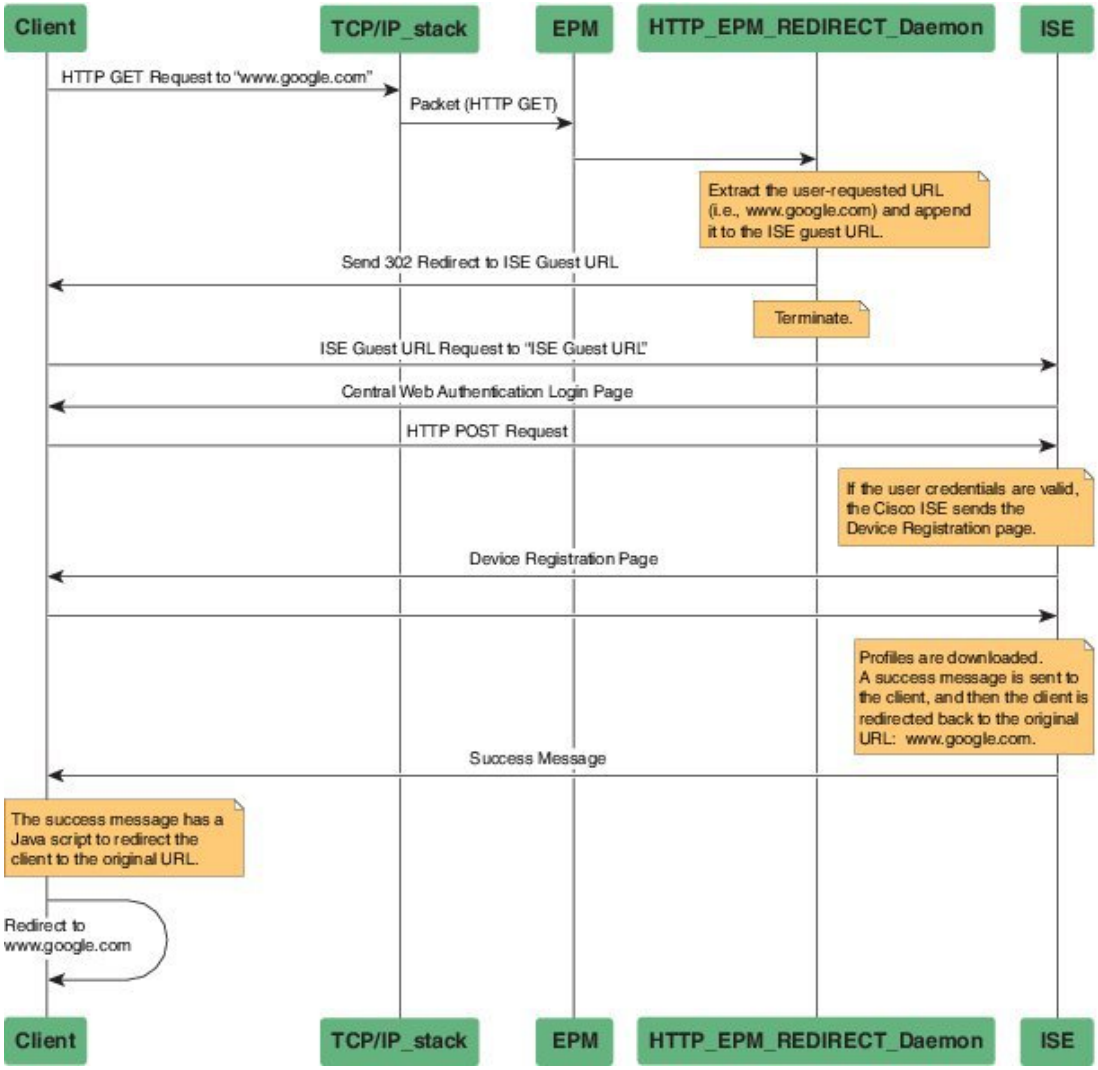
The following is an example of a redirect URL along with the original requested URL:

```
https://10.64.67.92:8443/guestportal/gateway?sessionId=0920269E0000000B0002426B&action=cwa&redirect_
url=http://www.cisco.com/
```

In this example, the URL, https://10.64.67.92:8443/guestportal/gateway?sessionId=0920269E0000000B0002426B&action=cwa is the URL for the guest portal, “&” tells the browser that what follows is a list of name value pairs, and redirect_url=http://www.cisco.com identifies the URL that the user originally requested and to which the user is redirected after completing the guest workflow.

This illustration displays the packet flow that redirects a user to the originally requested URL:

Figure 120: Original URL Redirection Packet Flow



1. A user accesses a network for the first time and sends an HTTP request to access www.google.com. When the user first accesses the network, a MAC authentication bypass (MAB) is triggered and the MAC address is sent to the Cisco ISE.
2. The Cisco ISE returns a RADIUS access-accept message (even if the MAC address is not received) along with the redirect access control list (ACL), the ACL-WEBAUTH-REDIRECT message, and the guest web portal URL to the device.

   The RADIUS message instructs the device to open a port that is restricted based on the configured port and the redirect ACLs, for regular network traffic.
3. When the user launches a web browser, the device intercepts the HTTP traffic and redirects the browser to the Cisco ISE central web authentication (CWA) guest web portal URL; the user-requested URL is extracted and appended to the Cisco ISE guest URL.

4. When the user is authenticated, the Cisco ISE sends the Device Registration page to the user. The user enters the required information, and the page is returned to the Cisco ISE. The Cisco ISE downloads user profiles and redirects the user to the originally requested URL <www.google.com>.

CHAPTER 112

Port-Based Traffic Control

Port-based traffic control is a set of Layer 2 features on the Cisco devices used to filter or block packets at the port level in response to specific traffic conditions. The following port-based traffic control features are supported:

- Storm Control
- Protected Ports
- Port Blocking

Information About Port-Based Traffic Control

Storm Control

Storm control prevents traffic on a LAN from being disrupted by a broadcast, multicast, or unicast storm on one of the physical interfaces. A LAN storm occurs when packets flood the LAN, creating excessive traffic and degrading network performance. Errors in the protocol-stack implementation, mistakes in network configurations, or users issuing a denial-of-service attack can cause a storm.

Storm control (or traffic suppression) monitors packets passing from an interface to the switching bus and determines if the packet is unicast, multicast, or broadcast. The switch counts the number of packets of a specified type received within the 1-second time interval and compares the measurement with a predefined suppression-level threshold.

Measured Traffic Activity

Storm control uses one of these methods to measure traffic activity:

- Bandwidth as a percentage of the total available bandwidth of the port that can be used by the broadcast, multicast, or unicast traffic
- Traffic rate in packets per second at which broadcast, multicast, or unicast packets are received
- Traffic rate in bits per second at which broadcast, multicast, or unicast packets are received

With each method, the port blocks traffic when the rising threshold is reached. The port remains blocked until the traffic rate drops below the falling threshold (if one is specified) and then resumes normal forwarding. If the falling suppression level is not specified, the device blocks all traffic until the traffic rate drops below the rising suppression level. In general, the higher the level, the less effective the protection against broadcast storms.

Note When the storm control threshold for multicast traffic is reached, all multicast traffic except control traffic, such as bridge protocol data unit (BDPU) and Cisco Discovery Protocol frames, are blocked. However, the device does not differentiate between routing updates, such as OSPF, and regular multicast data traffic, so both types of traffic are blocked.

Storm control for unicast is a combination of known unicast and unknown unicast traffic. When storm control for unicast is configured, and it exceeds the configured value, the storm will hit each type of traffic through the hardware policer. The following example describes how the unicast traffic is filtered, when the configured storm is 10%:

- Incoming traffic is unknown unicast 8% + known unicast 7%. Total of 15% storm is not filtered in hardware by the hardware policer.
- Incoming traffic is unknown unicast 11% + known unicast 7%. Total of 18% storm will hit unknown unicast traffic type, and the hardware policer will filter unknown traffic that exceeds 11%.
- Incoming traffic is unknown unicast 11% + known unicast 11%. Total of 22% storm will hit unknown unicast traffic and known unicast traffic, and the hardware policer will filter both unknown and unknown unicast traffic.

Note Do not configure both **storm-control unicast** and **storm-control unknown unicast** commands on an interface. If you configure both these commands, it might result in the unknown unicast storm control values to be modified in the hardware.

Traffic Patterns

Broadcast traffic being forwarded exceeded the configured threshold between time intervals T1 and T2 and between T4 and T5. When the amount of specified traffic exceeds the threshold, all traffic of that kind is dropped for the next time period. Therefore, broadcast traffic is blocked during the intervals following T2 and T5. At the next time interval (for example, T3), if broadcast traffic does not exceed the threshold, it is again forwarded.

The combination of the storm-control suppression level and the 1-second time interval controls the way the storm control algorithm works. A higher threshold allows more packets to pass through. A threshold value of 100 percent means that no limit is placed on the traffic. A value of 0.0 means that all broadcast, multicast, or unicast traffic on that port is blocked.

Note Because packets do not arrive at uniform intervals, the 1-second time interval during which traffic activity is measured can affect the behavior of storm control.

You use the **storm-control** interface configuration commands to set the threshold value for each traffic type.

Storm Control Using a Hardware Rate Limiter

Traffic storm control monitors incoming traffic levels over a configured interval. However, the reaction time taken by storm control is slightly slower as it is based on statistics counters to identify a storm. With the hardware rate limiter, the action is taken at the ASIC level, and as a result, the storm control action starts immediately; as soon as the traffic rate reaches the set threshold level. The hardware rate limiter implements policers for broadcast, multicast, unicast, and unknown unicast traffic.

Protected Ports

Some applications require that no traffic be forwarded at Layer 2 between ports on the same device so that one neighbor does not see the traffic generated by another neighbor. In such an environment, the use of protected ports ensures that there is no exchange of unicast, broadcast, or multicast traffic between these ports on the device.

Protected ports have these features:

- A protected port does not forward any traffic (unicast, multicast, or broadcast) to any other port that is also a protected port. Data traffic cannot be forwarded between protected ports at Layer 2; only control traffic, such as PIM packets, is forwarded because these packets are processed by the CPU and forwarded in software. All data traffic passing between protected ports must be forwarded through a Layer 3 device.
- Forwarding behavior between a protected port and a nonprotected port proceeds as usual.

Protected Ports Guidelines

You can configure protected ports on a physical interface (for example, Gigabit Ethernet port 1) or an EtherChannel group (for example, port-channel 5). When you enable protected ports for a port channel, it is enabled for all ports in the port-channel group.

By default no protected ports are defined.

Port Blocking

By default, the device floods packets with unknown destination MAC addresses out of all ports. If unknown unicast and multicast traffic is forwarded to a protected port, there could be security issues. To prevent unknown unicast or multicast traffic from being forwarded from one port to another, you can block a port (protected or nonprotected) from flooding unknown unicast or multicast packets to other ports.

Note With multicast traffic, the port blocking feature blocks only pure Layer 2 packets. Multicast packets that contain IPv4 or IPv6 information in the header are not blocked.

How to Configure Port-Based Traffic Control

Configuring Storm Control and Threshold Levels

You configure storm control on a port and enter the threshold level that you want to be used for a particular type of traffic.

However, because of hardware limitations and the way in which packets of different sizes are counted, threshold percentages are approximations. Depending on the sizes of the packets making up the incoming traffic, the actual enforced threshold might differ from the configured level by several percentage points.

Note Storm control is supported on physical interfaces. You can also configure storm control on an EtherChannel. When storm control is configured on an EtherChannel, the storm control settings propagate to the EtherChannel physical interfaces.

Follow these steps to storm control and threshold levels:

Before you begin

Storm control is supported on physical interfaces. You can also configure storm control on an EtherChannel. When storm control is configured on an EtherChannel, the storm control settings propagate to the EtherChannel physical interfaces.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface to be configured, and enters interface configuration mode.
Step 4	**storm-control** {**broadcast** | **multicast** | **unicast**} **level** {*level* [*level-low*] | **bps** *bps* [*bps-low*] | **pps** *pps* [*pps-low*]}	Configures broadcast, multicast, or unicast storm control. By default, storm control is disabled.

	Command or Action	Purpose
	Example: `Device(config-if)# storm-control unicast level 87 65`	• For *level*, specifies the rising threshold level for broadcast, multicast, or unicast traffic as a percentage (up to two decimal places) of the bandwidth. The port blocks traffic when the rising threshold is reached. The range is 0.00 to 100.00. • (Optional) For *level-low*, specifies the falling threshold level as a percentage (up to two decimal places) of the bandwidth. This value must be less than or equal to the rising suppression value. The port forwards traffic when traffic drops below this level. If you do not configure a falling suppression level, it is set to the rising suppression level. The range is 0.00 to 100.00. If you set the threshold to the maximum value (100 percent), no limit is placed on the traffic. If you set the threshold to 0.0, all broadcast, multicast, and unicast traffic on that port is blocked. • For **bps** *bps*, specifies the rising threshold level for broadcast, multicast, or unicast traffic in bits per second (up to one decimal place). The port blocks traffic when the rising threshold is reached. The range is 0.0 to 10000000000.0. • (Optional) For *bps-low*, specifies the falling threshold level in bits per second (up to one decimal place). It can be less than or equal to the rising threshold level. The port forwards traffic when traffic drops below this level. The range is 0.0 to 10000000000.0. • For **pps** *pps*, specifies the rising threshold level for broadcast, multicast, or unicast traffic in packets per second (up to one decimal place). The port blocks traffic when the rising threshold is reached. The range is 0.0 to 10000000000.0. • (Optional) For *pps-low*, specifies the falling threshold level in packets per second (up to one decimal place). It can be less than or equal to the rising threshold level. The port forwards traffic when

	Command or Action	Purpose
		traffic drops below this level. The range is **0.0 to** 10000000000.0. For BPS and PPS settings, you can use metric suffixes such as k, m, and g for large number thresholds.
Step 5	**storm-control action** {**shutdown** | **trap**} **Example:** `Device(config-if)# storm-control action trap`	Specifies the action to be taken when a storm is detected. Once a storm is detected, the **shutdown** or **trap** action is applied on all the traffic. The default is to filter out the traffic and not to send traps. • Select the **shutdown** keyword to error-disable the port during a storm. • Select the **trap** keyword to generate an SNMP trap when a storm is detected.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 7	**show storm-control** [*interface-id*] [**broadcast** | **multicast** | **unicast**] **Example:** `Device# show storm-control gigabitethernet1/1 unicast`	Verifies the storm control suppression levels set on the interface for the specified traffic type. If you do not enter a traffic type, details for all traffic types (broadcast, multicast and unicast) are displayed.

Configuring a Protected Port

Before you begin

Protected ports are not pre-defined. This is the task to configure one.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be configured, and enters interface configuration mode.
Step 4	**switchport protected** **Example:** `Device(config-if)# switchport protected`	Configures the interface to be a protected port.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Monitoring Protected Ports

Table 115: Commands for Displaying Protected Port Settings

Command	Purpose
show interfaces [*interface-id*] **switchport**	Displays the administrative and operational status of all switching (nonrouting) ports or the specified port, including port blocking and port protection settings.

Blocking Flooded Traffic on an Interface

The interface can be a physical interface or an EtherChannel group. When you block multicast or unicast traffic for a port channel, it is blocked on all ports in the port-channel group.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be configured, and enter interface configuration mode.
Step 4	**switchport block multicast** **Example:** `Device(config-if)# switchport block multicast`	Blocks unknown multicast forwarding out of the port.
Step 5	**switchport block unicast** **Example:** `Device(config-if)# switchport block unicast`	Blocks unknown unicast forwarding out of the port.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Monitoring Port Blocking

Table 116: Commands for Displaying Port Blocking Settings

Command	Purpose
show interfaces [*interface-id*] **switchport**	Displays the administrative and operational status of all switching (nonrouting) ports or the specified port, including port blocking and port protection settings.

CHAPTER 113

Port Security

Prerequisites for Port Security

If you try to set the maximum value to a number less than the number of secure addresses already configured on an interface, the command is rejected.

Restrictions for Port Security

- The maximum number of secure MAC addresses that you can configure on a switch is set by the maximum number of available MAC addresses allowed in the system. This number is the total of available MAC addresses, including those used for other Layer 2 functions and any other secure MAC addresses configured on interfaces.
- Port Security is not supported on EtherChanel interfaces.
- Port Security is not supported on private VLAN ports.
- We recommend that you do not enable port security on an 802.1X authenticator interface.

  When port-security is disabled on a port, the 802.1X sessions on the port get removed, because the aging timer and inactivity type is still configured. To ensure that the 802.1X sessions are not removed, when disabling port-security, disable the aging timer and inactivity type by removing the following commands:

  - **switchport port-security aging time 1**
  - **switchport port-security aging type inactivity**

  If the inactivity timer is required, see the section "Enabling and Configuring Port Security Aging".

Information About Port Security

Port Security

You can use the port security feature to restrict input to an interface by limiting and identifying MAC addresses of the stations allowed to access the port. When you assign secure MAC addresses to a secure port, the port does not forward packets with source addresses outside the group of defined addresses. If you limit the number of secure MAC addresses to one and assign a single secure MAC address, the workstation attached to that port is assured the full bandwidth of the port.

If a port is configured as a secure port and the maximum number of secure MAC addresses is reached, when the MAC address of a station attempting to access the port is different from any of the identified secure MAC addresses, a security violation occurs. Also, if a station with a secure MAC address configured or learned on one secure port attempts to access another secure port, a violation is flagged.

Types of Secure MAC Addresses

The switch supports these types of secure MAC addresses:

- Static secure MAC addresses—These are manually configured by using the **switchport port-security mac-address** *mac-address* interface configuration command, stored in the address table, and added to the switch running configuration.
- Dynamic secure MAC addresses—These are dynamically configured, stored only in the address table, and removed when the switch restarts.
- Sticky secure MAC addresses—These can be dynamically learned or manually configured, stored in the address table, and added to the running configuration. If these addresses are saved in the configuration file, when the switch restarts, the interface does not need to dynamically reconfigure them.

Default MAC Address Table Settings

Table 117: Default Settings for the MAC Address

Feature	Default Setting
Aging time	300 seconds
Dynamic addresses	Automatically learned
Static addresses	None configured

MAC Address Table Creation

With multiple MAC addresses supported on all ports, you can connect any port on the device to other network devices. The device provides dynamic addressing by learning the source address of packets it receives on each port and adding the address and its associated port number to the address table. As devices are added or

removed from the network, the device updates the address table, adding new dynamic addresses and aging out those that are not in use.

The aging interval is globally configured. However, the device maintains an address table for each VLAN, and STP can accelerate the aging interval on a per-VLAN basis.

The device sends packets between any combination of ports, based on the destination address of the received packet. Using the MAC address table, the device forwards the packet only to the port associated with the destination address. If the destination address is on the port that sent the packet, the packet is filtered and not forwarded. The device always uses the store-and-forward method: complete packets are stored and checked for errors before transmission.

Sticky Secure MAC Addresses

You can configure an interface to convert the dynamic MAC addresses to sticky secure MAC addresses and to add them to the running configuration by enabling sticky learning. The interface converts all the dynamic secure MAC addresses, including those that were dynamically learned before sticky learning was enabled, to sticky secure MAC addresses. All sticky secure MAC addresses are added to the running configuration.

The sticky secure MAC addresses do not automatically become part of the configuration file, which is the startup configuration used each time the switch restarts. If you save the sticky secure MAC addresses in the configuration file, when the switch restarts, the interface does not need to relearn these addresses. If you do not save the sticky secure addresses, they are lost.

If sticky learning is disabled, the sticky secure MAC addresses are converted to dynamic secure addresses and are removed from the running configuration.

Security Violations

It is a security violation when one of these situations occurs:

- The maximum number of secure MAC addresses have been added to the address table, and a station whose MAC address is not in the address table attempts to access the interface.
- An address learned or configured on one secure interface is seen on another secure interface in the same VLAN.
- Running diagnostic tests with port security enabled.

You can configure the interface for one of three violation modes, based on the action to be taken if a violation occurs:

- protect—when the number of secure MAC addresses reaches the maximum limit allowed on the port, packets with unknown source addresses are dropped until you remove a sufficient number of secure MAC addresses to drop below the maximum value or increase the number of maximum allowable addresses. You are not notified that a security violation has occurred.

Note We do not recommend configuring the protect violation mode on a trunk port. The protect mode disables learning when any VLAN reaches its maximum limit, even if the port has not reached its maximum limit.

- restrict—when the number of secure MAC addresses reaches the maximum limit allowed on the port, packets with unknown source addresses are dropped until you remove a sufficient number of secure MAC addresses to drop below the maximum value or increase the number of maximum allowable addresses. In this mode, you are notified that a security violation has occurred. An SNMP trap is sent, a syslog message is logged, and the violation counter increments.
- shutdown—a port security violation causes the interface to become error-disabled and to shut down immediately, and the port LED turns off. When a secure port is in the error-disabled state, you can bring it out of this state by entering the **errdisable recovery cause** *psecure-violation* global configuration command, or you can manually re-enable it by entering the **shutdown** and **no shut down** interface configuration commands. This is the default mode.
- shutdown vlan—Use to set the security violation mode per-VLAN. In this mode, the VLAN is error disabled instead of the entire port when a violation occurs

Table 118: Security Violation Mode Actions

Violation Mode	Traffic is forwarded [11]	Sends SNMP trap	Sends syslog message	Displays error message [12]	Violation counter increments	Shuts down port
protect	No	No	No	No	No	No
restrict	No	Yes	Yes	No	Yes	No
shutdown	No	No	No	No	Yes	Yes
shutdown vlan	No	No	Yes	No	Yes	No [13]

[11] Packets with unknown source addresses are dropped until you remove a sufficient number of secure MAC addresses.
[12] The switch returns an error message if you manually configure an address that would cause a security violation.
[13] Shuts down only the VLAN on which the violation occurred.

Port Security Aging

You can use port security aging to set the aging time for all secure addresses on a port. Two types of aging are supported per port:

- Absolute—The secure addresses on the port are deleted after the specified aging time.
- Inactivity—The secure addresses on the port are deleted only if the secure addresses are inactive for the specified aging time.

Default Port Security Configuration

Table 119: Default Port Security Configuration

Feature	Default Setting
Port security	Disabled on a port.
Sticky address learning	Disabled.
Maximum number of secure MAC addresses per port	One address
Violation mode	Shutdown. The port shuts down when the maximum number of secure MAC addresses is exceeded.
Port security aging	Disabled. Aging time is 0. Static aging is disabled. Type is absolute.

Port Security Configuration Guidelines

The following guidelines are applicable during port security configuration:

- Port security can only be configured on static access ports or trunk ports. A secure port cannot be a dynamic access port.
- A secure port cannot be a destination port for Switched Port Analyzer (SPAN).
- Voice VLAN is only supported on access ports and not on trunk ports, even though the configuration is allowed.
- When you enable port security on an interface that is also configured with a voice VLAN, set the maximum allowed secure addresses on the port to two. When the port is connected to a Cisco IP phone, the IP phone requires one MAC address. The Cisco IP phone address is learned on the voice VLAN, but is not learned on the access VLAN. If you connect a single PC to the Cisco IP phone, no additional MAC addresses are required. If you connect more than one PC to the Cisco IP phone, you must configure enough secure addresses to allow one for each PC and one for the phone.
- When a trunk port configured with port security and assigned to an access VLAN for data traffic and to a voice VLAN for voice traffic, entering the **switchport voice** and **switchport priority extend** interface configuration commands has no effect.

  When a connected device uses the same MAC address to request an IP address for the access VLAN and then an IP address for the voice VLAN, only the access VLAN is assigned an IP address.
- When you enter a maximum secure address value for an interface, and the new value is greater than the previous value, the new value overwrites the previously configured value. If the new value is less than the previous value and the number of configured secure addresses on the interface exceeds the new value, the command is rejected.
- The switch does not support port security aging of sticky secure MAC addresses.

Table 120: Port Security Compatibility with Other Switch Features

Type of Port or Feature on Port	Compatible with Port Security
DTP [14] port [15]	No
Trunk port	Yes
Dynamic-access port [16]	No
Routed port	No
SPAN source port	Yes
SPAN destination port	No
EtherChannel	No
Tunneling port	Yes
Protected port	Yes
IEEE 802.1x port	Yes
Voice VLAN port [17]	Yes
IP source guard	Yes
Dynamic Address Resolution Protocol (ARP) inspection	Yes
Flex Links	Yes

[14] DTP=Dynamic Trunking Protocol
[15] A port configured with the **switchport mode dynamic** interface configuration command.
[16] A VLAN Query Protocol (VQP) port configured with the **switchport access vlan dynamic** interface configuration command.
[17] You must set the maximum allowed secure addresses on the port to two plus the maximum number of secure addresses allowed on the access VLAN.

Management Traffic Control

A device in a network allows traffic like SNMP, HTTP, HTTPS, Telnet, Secure Shell SSH and Netconf through any port with any IP address. Traffic flow from various interfaces to the local devices might decrease the security strength in a network.

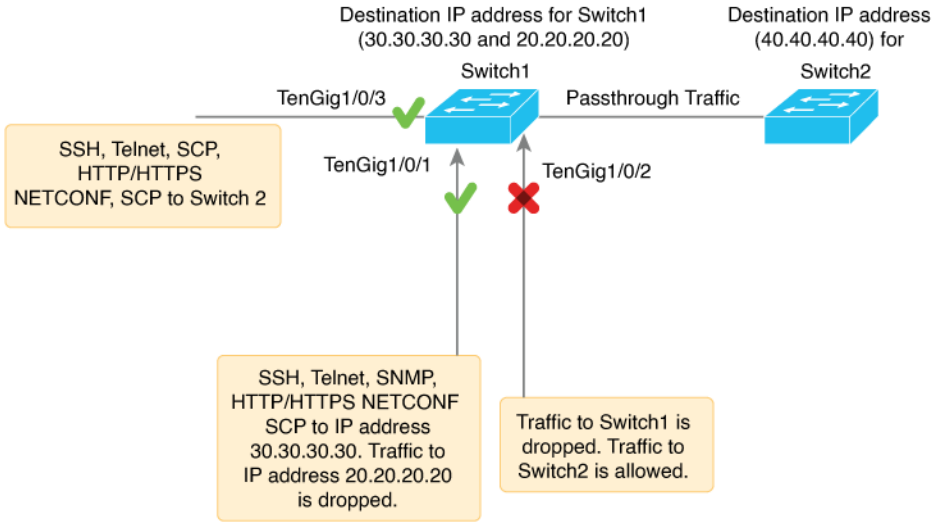
Management traffic control feature allows traffic to enter through a user-defined physical interface and restricts traffic to any other interfaces that is not defined by the user. When the feature is enabled a single IP address is assigned on the device to receive traffic. The user can configure the feature by defining an interface under the management traffic control feature. When the network protocol and the IP address is set according to the user's preference, traffic flow is allowed only through the defined interface.

The feature is supported on:

- Layer 2 physical interface.
- Layer 3 physical interface.

- Layer 2 port channel.
- Layer 3 port channel.
- App-hosting interface.

For example, in the following figure Switch1 and Switch2 are devices that are in a network. Management traffic control feature is enabled on Switch1 with the interface gigabitethernet 1/1 and destination IP address 30.30.30.30. Traffic is allowed through the interface with the enabled protocols SSH, Telnet, SNMP, HTTP, HTTPS, Netconf, SCP to destination IP address 30.30.30.30. Traffic passing through interface gigabitethernet 1/1 to IP address 20.20.20.20 is dropped. Management traffic control feature enables only one destination IP address to be configured. If interface gigabitethernet 1/2 is not defined by the management traffic control feature for any of the devices, traffic will not be allowed to Switch1 but can still pass through to its configured destination in the network.



Note

- You cannot configure IPv6 address configurations in the management traffic control feature.
- VRF based IP configuration is not supported on the managament traffic control feature.
- The management traffic control feature is not supported on interfaces like port-channel member, SVI, stack-port and management port.

Table 121: Supported protocols and the respective port numbers

Protocol	Keyword	Port Number
HTTPS	TCP	443
TELNET	TCP	23
SSH	TCP	22
NETCONF-SSH	TCP	830

Protocol	Keyword	Port Number
SNMP	UDP	161
HTTP	TCP	80

How to Configure Port Security

Enabling and Configuring Port Security

Before you begin

This task restricts input to an interface by limiting and identifying MAC addresses of the stations allowed to access the port:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface to be configured, and enter interface configuration mode.
Step 4	**switchport mode** {**access** | **trunk**} **Example:** `Device(config-if)# switchport mode access`	Sets the interface switchport mode as access or trunk; an interface in the default mode (dynamic auto) cannot be configured as a secure port.
Step 5	**switchport voice vlan** *vlan-id* **Example:** `Device(config-if)# switchport voice vlan`	Enables voice VLAN on a port. *vlan-id*—Specifies the VLAN to be used for voice traffic.

	Command or Action	Purpose
	22	
Step 6	**switchport port-security** **Example:** `Device(config-if)# switchport port-security`	Enables port security on the interface.
Step 7	**switchport port-security [maximum** *value* [**vlan** {*vlan-list* | {**access** | **voice**}}]] **Example:** `Device(config-if)# switchport port-security maximum 20`	(Optional) Sets the maximum number of secure MAC addresses for the interface. The maximum number of secure MAC addresses that you can configure on a switch is set by the maximum number of available MAC addresses allowed in the system. This number is the total of available MAC addresses, including those used for other Layer 2 functions and any other secure MAC addresses configured on interfaces. (Optional) **vlan**—sets a per-VLAN maximum value Enter one of these options after you enter the **vlan** keyword: • *vlan-list*—On a trunk port, you can set a per-VLAN maximum value on a range of VLANs separated by a hyphen or a series of VLANs separated by commas. For nonspecified VLANs, the per-VLAN maximum value is used. • **access**—On an access port, specifies the VLAN as an access VLAN. • **voice**—On an access port, specifies the VLAN as a voice VLAN. **Note** The **voice** keyword is available only if a voice VLAN is configured on a port and if that port is not the access VLAN. If an interface is configured for voice VLAN, configure a maximum of two secure MAC addresses.
Step 8	**switchport port-security violation** {**protect** | **restrict** | **shutdown** | **shutdown vlan**} **Example:** `Device(config-if)# switchport`	(Optional) Sets the violation mode, the action to be taken when a security violation is detected, as one of these: • **protect**—When the number of port secure MAC addresses reaches the maximum

	Command or Action	Purpose
	`port-security violation restrict`	limit allowed on the port, packets with unknown source addresses are dropped until you remove a sufficient number of secure MAC addresses to drop below the maximum value or increase the number of maximum allowable addresses. You are not notified that a security violation has occurred. **Note** We do not recommend configuring the protect mode on a trunk port. The protect mode disables learning when any VLAN reaches its maximum limit, even if the port has not reached its maximum limit. • **restrict**—When the number of secure MAC addresses reaches the limit allowed on the port, packets with unknown source addresses are dropped until you remove a sufficient number of secure MAC addresses or increase the number of maximum allowable addresses. An SNMP trap is sent, a syslog message is logged, and the violation counter increments. • **shutdown**—The interface is error-disabled when a violation occurs, and the port LED turns off. An SNMP trap is sent, a syslog message is logged, and the violation counter increments. • **shutdown vlan**—Use to set the security violation mode per VLAN. In this mode, the VLAN is error disabled instead of the entire port when a violation occurs. **Note** When a secure port is in the error-disabled state, you can bring it out of this state by entering the **errdisable recovery cause psecure-violation** global configuration command. You can manually re-enable it by entering the **shutdown** and **no shutdown** interface configuration commands or by using the **clear errdisable interface vlan** privileged EXEC command.
Step 9	**switchport port-security [mac-address** *mac-address* [**vlan** {*vlan-id* | {**access** | **voice**}}]	(Optional) Enters a secure MAC address for the interface. You can use this command to

	Command or Action	Purpose
	Example: `DEvice(config-if)# switchport port-security mac-address 00:A0:C7:12:C9:25 vlan 3 voice`	enter the maximum number of secure MAC addresses. If you configure fewer secure MAC addresses than the maximum, the remaining MAC addresses are dynamically learned. **Note** If you enable sticky learning after you enter this command, the secure addresses that were dynamically learned are converted to sticky secure MAC addresses and are added to the running configuration. (Optional) **vlan**—sets a per-VLAN maximum value. Enter one of these options after you enter the **vlan** keyword: • *vlan-id*—On a trunk port, you can specify the VLAN ID and the MAC address. If you do not specify a VLAN ID, the native VLAN is used. • **access**—On an access port, specifies the VLAN as an access VLAN. • **voice**—On an access port, specifies the VLAN as a voice VLAN. **Note** The **voice** keyword is available only if a voice VLAN is configured on a port and if that port is not the access VLAN. If an interface is configured for voice VLAN, configure a maximum of two secure MAC addresses.
Step 10	**switchport port-security mac-address sticky** **Example:** `Device(config-if)# switchport port-security mac-address sticky`	(Optional) Enables sticky learning on the interface.
Step 11	**switchport port-security mac-address sticky** [*mac-address* | **vlan** {*vlan-id* | {**access** | **voice**}}] **Example:** `Device(config-if)# switchport port-security mac-address sticky 00:A0:C7:12:C9:25 vlan voice`	(Optional) Enters a sticky secure MAC address, repeating the command as many times as necessary. If you configure fewer secure MAC addresses than the maximum, the remaining MAC addresses are dynamically learned, are converted to sticky secure MAC addresses, and are added to the running configuration. **Note**

	Command or Action	Purpose
		If you do not enable sticky learning before this command is entered, an error message appears, and you cannot enter a sticky secure MAC address. (Optional) **vlan**—sets a per-VLAN maximum value. Enter one of these options after you enter the **vlan** keyword: • *vlan-id*—On a trunk port, you can specify the VLAN ID and the MAC address. If you do not specify a VLAN ID, the native VLAN is used. • **access**—On an access port, specifies the VLAN as an access VLAN. • **voice**—On an access port, specifies the VLAN as a voice VLAN. **Note** The **voice** keyword is available only if a voice VLAN is configured on a port and if that port is not the access VLAN.
Step 12	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 13	**show port-security** **Example:** `Device# show port-security`	Displays information about the port-security settings.

Enabling and Configuring Port Security Aging

Use this feature to remove and add devices on a secure port without manually deleting the existing secure MAC addresses and to still limit the number of secure addresses on a port. You can enable or disable the aging of secure addresses on a per-port basis.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface to be configured, and enter interface configuration mode.
Step 4	**switchport port-security aging** {**static** | **time** *time* | **type** {**absolute** | **inactivity**}} **Example:** `Device(config-if)# switchport port-security aging time 120`	Enables or disable static aging for the secure port, or set the aging time or type. **Note** The switch does not support port security aging of sticky secure addresses. Enter **static** to enable aging for statically configured secure addresses on this port. For *time*, specifies the aging time for this port. The valid range is from 0 to 1440 minutes. For **type**, select one of these keywords: • **absolute**—Sets the aging type as absolute aging. All the secure addresses on this port age out exactly after the time (minutes) specified lapses and are removed from the secure address list. • **inactivity**—Sets the aging type as inactivity aging. The secure addresses on this port age out only if there is no data traffic from the secure source addresses for the specified time period.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 6	**show port-security** [**interface** *interface-id*] [**address**] **Example:**	Displays information about the port-security settings on the specified interface.

	Command or Action	Purpose
	`Device# show port-security interface gigabitethernet1/1`	

Changing the Address Aging Time

Follow these steps to configure the dynamic address table aging time:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mac address-table aging-time** [*0* | *10-1000000*] [**routed-mac** | **vlan** *vlan-id*] **Example:** `Device(config)# mac address-table aging-time 500 vlan 2`	Sets the length of time that a dynamic entry remains in the MAC address table after the entry is used or updated. The range is 10 to 1000000 seconds. The default is 300. You can also enter 0, which disables aging. Static address entries are never aged or removed from the table. *vlan-id*—Valid IDs are 1 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Monitoring Port Security

Table 122: Commands for Displaying Port Security Status and Configuration

Command	Purpose
show port-security [**interface** *interface-id*]	Displays port security settings for the device or for the specified interface, including the maximum allowed number of secure MAC addresses for each interface, the number of secure MAC addresses on the interface, the number of security violations that have occurred, and the violation mode.
show port-security [**interface** *interface-id*] **address**	Displays all secure MAC addresses configured on all device interfaces or on a specified interface with aging information for each address.
show port-security interface *interface-id* **vlan**	Displays the number of secure MAC addresses configured per VLAN on the specified interface.

Configuring Management Traffic Control

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mgmt-traffic control ipv4** **Example:** `Device# mgmt-traffic control ipv4` `Device(config-mtc-ipv4)#`	Enables the management traffic control feature.
Step 4	**interface** *interface-id* **Example:** `Device(config-mtc-ipv4)# interface gigabitethernet1/1` `Device(config-mtc-ipv4)#`	Defines the interface through which traffic is allowed.

	Command or Action	Purpose
Step 5	**protocol** { [**telnet http https netconf scp snmp ssh**] } **Example:** `Device(config-mtc-ipv4)# protocol telnet http https netconf scp snmp ssh` `Device(config-mtc-ipv4)#`	Enables the specified network protocols in the interface.
Step 6	**address** *ip-address* **Example:** `Device(config-mtc-ipv4)# address 30.30.30.30` `Device(config-mtc-ipv4)#`	Specifies the destination address of traffic.
Step 7	**end** **Example:** `Device(config-mtc-ipv4)# end`	Exits management traffic control configuration mode and returns to privileged EXEC mode.

Configuration Examples for Port Security

This example shows how to enable port security on a port and to set the maximum number of secure addresses to 50. The violation mode is the default, no static secure MAC addresses are configured, and sticky learning is enabled.

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet1/1
Device(config-if)# switchport mode access
Device(config-if)# switchport port-security
Device(config-if)# switchport port-security maximum 50
Device(config-if)# switchport port-security mac-address sticky
Device(config-if)# end
```

This example shows how to configure a static secure MAC address on VLAN 3 on a port:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet1/1
Device(config-if)# switchport mode trunk
Device(config-if)# switchport port-security
Device(config-if)# switchport port-security mac-address 0000.0200.0004 vlan 3
Device(config-if)# end
```

This example shows how to enable sticky port security on a port, to manually configure MAC addresses for data VLAN and voice VLAN, and to set the total maximum number of secure addresses to 20 (10 for data VLAN and 10 for voice VLAN).

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet1/1
Device(config-if)# switchport access vlan 21
```

```
Device(config-if)# switchport mode access
Device(config-if)# switchport voice vlan 22
Device(config-if)# switchport port-security
Device(config-if)# switchport port-security maximum 20
Device(config-if)# switchport port-security violation restrict
Device(config-if)# switchport port-security mac-address sticky
Device(config-if)# switchport port-security mac-address sticky 0000.0000.0002
Device(config-if)# switchport port-security mac-address 0000.0000.0003
Device(config-if)# switchport port-security mac-address sticky 0000.0000.0001 vlan voice
Device(config-if)# switchport port-security mac-address 0000.0000.0004 vlan voice
Device(config-if)# switchport port-security maximum 10 vlan access
Device(config-if)# switchport port-security maximum 10 vlan voice
Device(config-if)# end
```

CHAPTER 114

Configuring Control Plane Policing

Restrictions for Control Plane Policing

Restrictions for control plane policing (CoPP) include the following:

- Only ingress CoPP is supported. The **system-cpp-policy** policy-map is available on the control plane interface, and only in the ingress direction.
- Only the **system-cpp-policy** policy-map can be installed on the control plane interface.
- The **system-cpp-policy** policy-map and the system-defined classes cannot be modified or deleted.
- Only the **police** action is allowed under the **system-cpp-policy** policy-map. The police rate for system-defined classes must be configured only in packets per second (pps).
- One or more CPU queues are part of each class-map. Where multiple CPU queues belong to one class-map, changing the policer rate of a class-map affects all CPU queues that belong to that class-map. Similarly, disabling the policer in a class-map disables all queues that belong to that class-map. See *Table: System-Defined Values for CoPP* for information about which CPU queues belong to each class-map.
- We recommend not disabling the policer for a system-defined class map, that is, do not configure **no police rate** *rate* **pps** command. Doing so affects the overall system health in case of high traffic towards the CPU. Further, even if you disable the policer rate for a system-defined class map, the systems automatically reverts to the default policer rate after system bootup in order to protect the system bring-up process.
- The **show run** command does not display information about classes configured under `system-cpp policy`, when they are left at default values. Use the **show policy-map system-cpp-policy** or the **show policy-map control-plane** commands instead.

  You can continue use the **show run** command to display information about custom policies.
- A protocol with a large number of CPU-bound packets may impact other protocols in the same class, as some of these protocols share the same policer. For example, Address Resolution Protocol (ARP) shares

4000 hardware policers with an array of host protocols like Telnet, Internet Control Message Protocol (ICMP), SSH, FTP, and SNMP in the system-cpp-police-forus class. If there is an ARP poisoning or an ICMP attack, hardware policers start throttling any incoming traffic that exceeds 4000 packets per second to protect the CPU and the overall integrity of the system. As a result, ARP and ICMP host protocols are dropped, along with any other host protocols that share the same class.

- The creation of user-defined class-maps is not supported.

Information About Control Plane Policing

This chapter describes how CoPP works on your device and how to configure it.

Overview of Control Plane Policing

The CoPP feature improves security on your device by protecting the CPU from unnecessary traffic and denial of service (DoS) attacks. It can also protect control traffic and management traffic from traffic drops caused by high volumes of other, lower priority traffic.

Your device is typically segmented into three planes of operation, each with its own objective:

- The data plane, to forward data packets.
- The control plane, to route data correctly.
- The management plane, to manage network elements.

You can use CoPP to protect most of the CPU-bound traffic and ensure routing stability, reachability, and packet delivery. Most importantly, you can use CoPP to protect the CPU from a DoS attack.

CoPP uses the modular QoS command-line interface (MQC) and CPU queues to achieve these objectives. Different types of control plane traffic are grouped together based on certain criteria, and assigned to a CPU queue. You can manage these CPU queues by configuring dedicated policers in hardware. For example, you can modify the policer rate for certain CPU queues (traffic-type), or you can disable the policer for a certain type of traffic.

Although the policers are configured in hardware, CoPP does not affect CPU performance or the performance of the data plane. But since it limits the number of packets going to CPU, the CPU load is controlled. This means that services waiting for packets from hardware may see a more controlled rate of incoming packets (the rate being user-configurable).

System-Defined Aspects of Control Plane Policing

When you power-up the device for the first time, the system automatically performs the following tasks:

- Looks for policy-map **system-cpp-policy**. If not found, the system creates and installs it on the control-plane.
- Creates eighteen class-maps under **system-cpp-policy**.

  The next time you power-up the device, the system detects the policy and class maps that have already been created.

- Enables all CPU queues by default, with their respective default rate. The default rates are indicated in the table System-Defined Values for CoPP.

The **system-cpp-policy** policy map is a system-default policy map, and normally, you do not have to expressly save it to the startup configuration of the device. But, a *failed* bulk synchronization with a standby device can result in the configuration being erased from the startup configuration. In case this happens, you have to manually save the **system-cpp-policy** policy map to the startup configuration. Use the **show running-config** privileged EXEC command to verify that it has been saved.

The following table (System-Defined Values for CoPP) lists the class-maps that the system creates when you load the device. It lists the policer that corresponds to each class-map and one or more CPU queues that are grouped under each class-map. There is a one-to-one mapping of class-maps to policers; and one or more CPU queues map to a class-map. This is followed by another table (CPU Queues and Associated Features), which lists features associated with each CPU queue.

Table 123: System-Defined Values for CoPP

Class Maps Names	Policer Index (Policer No.)	CPU queues (Queue No.)
system-cpp- police-data	WK_CPP_POLICE_DATA(0)	WK_CPU_Q_ICMP_GEN(3) WK_CPU_Q_BROADCAST(12) WK_CPU_Q_ICMP_REDIRECT(6)
system-cpp-police-l2- control	WK_CPP_POLICE_L2_ CONTROL(1)	WK_CPU_Q_L2_CONTROL(1)
system-cpp-police-routing-control	WK_CPP_POLICE_ROUTING_CONTROL(2)	WK_CPU_Q_ROUTING_CONTROL(4) WK_CPU_Q_LOW_LATENCY (27)
system-cpp-police-punt-webauth	WK_CPP_POLICE_PU NT_WEBAUTH(7)	WK_CPU_Q_PUNT_WEBAUTH(22)
system-cpp-police- topology-control	WK_CPP_POLICE_TOPOLOGY_CONTROL(8)	WK_CPU_Q_TOPOLOGY_CONTROL(15)
system-cpp-police- multicast	WK_CPP_POLICE_MULTICAST(9)	WK_CPU_Q_TRANSIT_TRAFFIC(18) WK_CPU_Q_MCAST_DATA(30)
system-cpp-police-sys- data	WK_CPP_POLICE_SYS _DATA(10)	WK_CPU_Q_OPENFLOW (13) WK_CPU_Q_CRYPTO_CONTROL(23) WK_CPU_Q_EXCEPTION(24) WK_CPU_Q_EGR_EXCEPTION(28) WK_CPU_Q_NFL_SAMPLED_DATA(26) WK_CPU_Q_GOLD_PKT(31) WK_CPU_Q_RPF_FAILED(19)
system-cpp-police-dot1x-auth	WK_CPP_POLICE_DOT1X(11)	WK_CPU_Q_DOT1X_AUTH(0)
system-cpp-police- protocol-snooping	WK_CPP_POLICE_PR(12)	WK_CPU_Q_PROTO_SNOOPING(16)

Class Maps Names	Policer Index (Policer No.)	CPU queues (Queue No.)
system-cpp-police-dhcp-snooping	WK_CPP_DHCP_SNOOPING(6)	WK_CPU_Q_DHCP_SNOOPING(17)
system-cpp-police-sw-forward	WK_CPP_POLICE_SW_FWD (13)	WK_CPU_Q_SW_FORWARDING_Q(14) WK_CPU_Q_LOGGING(21) WK_CPU_Q_L2_LVX_DATA_PACK (11)
system-cpp-police-forus	WK_CPP_POLICE_FORUS(14)	WK_CPU_Q_FORUS_ADDR_RESOLUTION(5) WK_CPU_Q_FORUS_TRAFFIC(2)
system-cpp-police- multicast-end-station	WK_CPP_POLICE_MULTICAST_SNOOPING(15)	WK_CPU_Q_MCAST_END_STA TION_SERVICE(20)
system-cpp-default	WK_CPP_POLICE_DEFAULT_POLICER(16)	WK_CPU_Q_INTER_FED_TRAFFIC(7) WK_CPU_Q_EWLC_CONTROL(9) WK_CPU_Q_EWLC_DATA(10)
system-cpp-police-stackwise-virt-control	WK_CPP_STACKWISE_VIRTUAL_CONTROL(5)	WK_CPU_Q_STACKWISE_VIRTUAL_CONTROL (29)
system-cpp-police-l2lvx-control	WK_CPP_ L2_LVX_CONT_PACK(4)	WK_CPU_Q_L2_LVX_CONT_PACK(8)
system-cpp-police-high-rate-app	WK_CPP_HIGH_RATE_APP(18)	WK_CPU_Q_HIGH_RATE_APP(23)
system-cpp-police-system-critical	WK_CPP_SYSTEM_CRITICAL(3)	WK_CPU_Q_SYSTEM_CRITICAL(25)

The following table lists the CPU queues and the feature(s) associated with each CPU queue.

Table 124: CPU Queues and Associated Features

CPU queues (Queue No.)	Feature(s)
WK_CPU_Q_DOT1X_AUTH(0)	IEEE 802.1x Port-Based Authentication

CPU queues (Queue No.)	Feature(s)
WK_CPU_Q_L2_CONTROL(1)	Dynamic Trunking Protocol (DTP) VLAN Trunking Protocol (VTP) Port Aggregation Protocol (PAgP) Client Information Signaling Protocol (CISP) Message session relay protocol Multiple VLAN Registration Protocol (MVRP) Metropolitan Mobile Network (MMN) Link Level Discovery Protocol (LLDP) UniDirectional Link Detection (UDLD) Link Aggregation Control Protocol (LACP) Cisco Discovery Protocol (CDP) Spanning Tree Protocol (STP)
WK_CPU_Q_FORUS_TRAFFIC(2)	Host such as Telnet,Pingv4 and Pingv6, and SNMP Keepalive / loopback detection Initiate-Internet Key Exchange (IKE) protocol (IPSec)
WK_CPU_Q_ICMP_GEN(3)	ICMP - destination unreachable ICMP-TTL expired

CPU queues (Queue No.)	Feature(s)
WK_CPU_Q_ROUTING_CONTROL(4)	Routing Information Protocol version 1 (RIPv1) RIPv2 Interior Gateway Routing Protocol (IGRP) Border Gateway Protocol (BGP) PIM-UDP Virtual Router Redundancy Protocol (VRRP) Hot Standby Router Protocol version 1 (HSRPv1) HSRPv2 Gateway Load Balancing Protocol (GLBP) Label Distribution Protocol (LDP) Web Cache Communication Protocol (WCCP) Routing Information Protocol next generation (RIPng) Open Shortest Path First (OSPF) Open Shortest Path First version 3(OSPFv3) Enhanced Interior Gateway Routing Protocol (EIGRP) Enhanced Interior Gateway Routing Protocol version 6 (EIGRPv6) DHCPv6 Protocol Independent Multicast (PIM) Protocol Independent Multicast version 6 (PIMv6) Hot Standby Router Protocol next generation (HSRPng) IPv6 control Generic Routing Encapsulation (GRE) keepalive Network Address Translation (NAT) punt Intermediate System-to-Intermediate System (IS-IS)
WK_CPU_Q_FORUS_ADDR_RESOLUTION(5)	Address Resolution Protocol (ARP) IPv6 neighbor advertisement and neighbor solicitation
WK_CPU_Q_ICMP_REDIRECT(6)	Internet Control Message Protocol (ICMP) redirect
WK_CPU_Q_INTER_FED_TRAFFIC(7)	Layer 2 bridge domain inject for internal communication.
WK_CPU_Q_L2_LVX_CONT_PACK(8)	Exchange ID (XID) packet

CPU queues (Queue No.)	Feature(s)
WK_CPU_Q_EWLC_CONTROL(9)	Embedded Wirelss Controller (eWLC) [Control and Provisioning of Wireless Access Points (CAPWAP) (UDP 5246)]
WK_CPU_Q_EWLC_DATA(10)	eWLC data packet (CAPWAP DATA, UDP 5247)
WK_CPU_Q_L2_LVX_DATA_PACK(11)	Unknown unicast packet punted for map request.
WK_CPU_Q_BROADCAST(12)	All types of broadcast
WK_CPU_Q_OPENFLOW(13)	Learning cache overflow (Layer 2 + Layer 3)
WK_CPU_Q_CONTROLLER_PUNT(14)	Data - access control list (ACL) Full Data - IPv4 options Data - IPv6 hop-by-hop Data - out-of-resources / catch all Data - Reverse Path Forwarding (RPF) incomplete Glean packet
WK_CPU_Q_TOPOLOGY_CONTROL(15)	Spanning Tree Protocol (STP) Resilient Ethernet Protocol (REP) Shared Spanning Tree Protocol (SSTP)
WK_CPU_Q_PROTO_SNOOPING(16)	Address Resolution Protocol (ARP) snooping for Dynamic ARP Inspection (DAI)
WK_CPU_Q_DHCP_SNOOPING(17)	DHCP snooping
WK_CPU_Q_TRANSIT_TRAFFIC(18)	This is used for packets punted by NAT, which need to be handled in the software path.
WK_CPU_Q_RPF_FAILED(19)	Data – mRPF (multicast RPF) failed
WK_CPU_Q_MCAST_END_STATION_SERVICE(20)	Internet Group Management Protocol (IGMP) / Multicast Listener Discovery (MLD) control
WK_CPU_Q_LOGGING(21)	Access control list (ACL) logging
WK_CPU_Q_PUNT_WEBAUTH(22)	Web Authentication
WK_CPU_Q_HIGH_RATE_APP(23)	Wired Application Visibility and Control (WDAVC) traffic Network-Based Application Recognition (NBAR) traffic Encrypted Traffic Analytics (ETA) for traffic analysis and classification.

CPU queues (Queue No.)	Feature(s)
WK_CPU_Q_EXCEPTION(24)	IKE indication IP learning violation IP port security violation IP Static address violation IPv6 scope check Remote Copy Protocol (RCP) exception Unicast RPF fail
WK_CPU_Q_SYSTEM_CRITICAL(25)	Media Signaling/ Wireless Proxy ARP
WK_CPU_Q_NFL_SAMPLED_DATA(26)	Netflow sampled data and Media Services Proxy (MSP)
WK_CPU_Q_LOW_LATENCY(27)	Bidirectional Forwarding Detection (BFD), Precision Time Protocol (PTP)
WK_CPU_Q_EGR_EXCEPTION(28)	Egress resolution exception
WK_CPU_Q_MCAST_DATA(29)	Data - (S,G) creation Data - local joins Data - PIM Registration Data - SPT switchover Data - Multicast
WK_CPU_Q_GOLD_PKT(30)	Gold

User-Configurable Aspects of Control Plane Policing

You can perform these tasks to manage control plane traffic:

Note All system-cpp-policy configurations must be saved so they are retained after reboot.

Enable or Disable a Policer for CPU Queues

Enable a policer for a CPU queue, by configuring a policer action (in packets per second) under the corresponding class-map, within the system-cpp-policy policy-map.

Disable a policer for CPU queue, by removing the policer action under the corresponding class-map, within the system-cpp-policy policy-map.

Note If a default policer is already present, carefully consider and control its removal; otherwise the system may see a CPU hog or other anomalies, such as control packet drops.

Change the Policer Rate

You can do this by configuring a policer rate action (in packets per second), under the corresponding class-map, within thesystem-cpp-policy policy-map.

When setting a policer rate, note that the rate you set is automatically converted to the nearest multiple of 200. For instance, if you set the policer rate of a CPU queue 100 pps, the system changes it to 200; or if set the policer rate to 650, the system changes it to 600. See Example: Setting the Default Policer Rates for All CPU Queues in this chapter, for sample output that displays this behavior

Set Policer Rates to Default

Set the policer for CPU queues to their default values, by entering the **cpp system-default** command in global configuration mode.

Upgrading or Downgrading the Software Version

Software Version Upgrades and CoPP

When you upgrade the software version on your device, the system checks and make the necessary updates as required for CoPP (For instance, it checks for the `system-cpp-policy` policy map and creates it if missing). You may also have to complete certain tasks before or after the upgrade activity. This is to ensure that any configuration updates are reflected correctly and CoPP continues to work as expected. Depending on the method you use to upgrade the software, upgrade-related tasks may be optional or recommended in some scenarios, and mandatory in others.

The system actions and user actions for an upgrade, are described here. Also included, are any release-specfic caveats.

System Actions for an Upgrade

When you upgrade the software version on your device, the system performs these actions. This applies to all upgrade methods:

- If the device did not have a `system-cpp-policy` policy map before upgrade, then on upgrade, the system creates a default policy map.
- If the device had a `system-cpp-policy` policy map before upgrade, then on upgrade, the system does not re-generate the policy.

User Actions for an Upgrade

User actions for an upgrade – depending on upgrade method:

Upgrade Method	Condition	Action Time and Action	Purpose
Regular[18]	None	**After upgrade (required)** Enter the **cpp system-default** command in global configuration mode	To get the latest, default policer rates.

[18] Refers to a software upgrade method that involves a reload of the switch. Can be install or bundle mode.

Software Version Downgrades and CoPP

The system actions and user actions for a downgrade, are described here.

System Actions for a Downgrade

When you downgrade the software version on your device, the system performs these actions. This applies to all downgrade methods:

- The system retains the `system-cpp-policy` policy map on the device, and installs it on the control plane.

User Actions for a Downgrade

User actions for a downgrade:

Upgrade Method	Condition	Action Time and Action	Purpose
Regular[19]	None	No action required	Not applicable

[19] Refers to a software upgrade method that involves a reload of the switch. Can be install or bundle mode.

If you downgrade the software version and then upgrade, the system action and user actions that apply are the same as those mentioned for upgrades.

How to Configure CoPP

Enabling a CPU Queue and Changing the Policer Rate

The procedure to enable a CPU queue and change the policer rate of a CPU queue is the same. Follow these steps:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map system-cpp-policy` `Device(config-pmap)#`	Enters the policy map configuration mode.
Step 4	**class** *class-name* **Example:** `Device(config-pmap)# class system-cpp-police-protocol-snooping` `Device(config-pmap-c)#`	Enters the class action configuration mode. Enter the name of the class that corresponds to the CPU queue you want to enable. See table *System-Defined Values for CoPP*.
Step 5	**police rate** *rate* **pps** **Example:** `Device(config-pmap-c)# police rate 100 pps` `Device(config-pmap-c-police)#`	Specifies an upper limit on the number of incoming packets processed per second, for the specified traffic class. **Note** The rate you specify is applied to all CPU queues that belong to the class-map you have specified.
Step 6	exit **Example:** `Device(config-pmap-c-police)# exit` `Device(config-pmap-c)# exit` `Device(config-pmap)# exit` `Device(config)#`	Returns to the global configuration mode.
Step 7	**control-plane** **Example:** `Device(config)# control-plane` `Device(config-cp)#`	Enters the control plane (config-cp) configuration mode
Step 8	**service-policy input** *policy-name* **Example:** `Device(config)# control-plane` `Device(config-cp)#service-policy input system-cpp-policy` `Device(config-cp)#`	Installs system-cpp-policy in FED. This command is required for you to see the FED policy. Not configuring this command will lead to an error.

	Command or Action	Purpose
Step 9	**end** **Example:** `Device(config-cp)# end`	Returns to the privileged EXEC mode.
Step 10	**show policy-map control-plane** **Example:** `Device# show policy-map control-plane`	Displays all the classes configured under `system-cpp policy`, the rates configured for the various traffic types, and statistics

Disabling a CPU Queue

Follow these steps to disable a CPU queue:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map system-cpp-policy` `Device(config-pmap)#`	Enters the policy map configuration mode.
Step 4	**class** *class-name* **Example:** `Device(config-pmap)# class system-cpp-police-protocol-snooping` `Device(config-pmap-c)#`	Enters the class action configuration mode. Enter the name of the class that corresponds to the CPU queue you want to disable. See the table, *System-Defined Values for CoPP*.
Step 5	**no police rate** *rate* **pps** **Example:** `Device(config-pmap-c)# no police rate 100 pps`	Disables incoming packet processing for the specified traffic class. **Note** This disables all CPU queues that belong to the class-map you have specified.

	Command or Action	Purpose
Step 6	**end** **Example:** `Device(config-pmap-c)# end`	Returns to the privileged EXEC mode.
Step 7	**show policy-map control-plane** **Example:** `Device# show policy-map control-plane`	Displays all the classes configured under `system-cpp policy` and the rates configured for the various traffic types and statistics.

Setting the Default Policer Rates for All CPU Queues

Follow these steps to set the policer rates for all CPU queues to their default rates:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cpp system-default** **Example:** `Device(config)# cpp system-default` `Defaulting CPP : Policer rate for all classes will be set to their defaults`	Sets the policer rates for all the classes to the default rate.
Step 4	**end** **Example:** `Device(config)# end`	Returns to the privileged EXEC mode.
Step 5	**show platform hardware fed switch** { *switch-number* } **qos que stats internal cpu policer** **Example:**	Displays the rates configured for the various traffic types.

	Command or Action	Purpose
	`Device# show platform hardware fed switch 1 qos que stat internal cpu policer`	

Configuration Examples for Control Plane Policing

Example: Enabling and Changing the Policer Rate of a CPU Queue

This example shows how to enable a CPU queue or to change the policer rate of a CPU queue. Here the **class system-cpp-police-protocol-snooping** CPU queue is enabled with the policer rate of **2000 pps** .

```
Device> enable
Device# configure terminal
Device(config)# policy-map system-cpp-policy
Device(config-pmap)# class system-cpp-police-protocol-snooping
Device(config-pmap-c)# police rate 2000 pps
Device(config-pmap-c-police)# end


Device# show policy-map control-plane

Control Plane

  Service-policy input: system-cpp-policy

    <output truncated>


    Class-map: system-cpp-police-dot1x-auth (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: none
      police:
          rate 1000 pps, burst 244 packets
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          drop

      <output truncated>

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Example: Disabling a CPU Queue

This example shows how to disable a CPU queue. Here the **class system-cpp-police-protocol-snooping** CPU queue is disabled.

```
Device> enable
Device# configure terminal
Device(config)# policy-map system-cpp-policy
Device(config-pmap)# class system-cpp-police-protocol-snooping
Device(config-pmap-c)# no police rate 100 pps
Device(config-pmap-c)# end


Device# show running-config | begin system-cpp-policy

policy-map system-cpp-policy
 class system-cpp-police-data
  police rate 200 pps
 class system-cpp-police-sys-data
  police rate 100 pps
 class system-cpp-police-sw-forward
  police rate 1000 pps
 class system-cpp-police-multicast
  police rate 500 pps
 class system-cpp-police-multicast-end-station
  police rate 2000 pps
 class system-cpp-police-punt-webauth
 class system-cpp-police-l2-control
 class system-cpp-police-routing-control
  police rate 500 pps
 class system-cpp-police-control-low-priority
 class system-cpp-police-wireless-priority1
 class system-cpp-police-wireless-priority2
 class system-cpp-police-wireless-priority3-4-5
 class system-cpp-police-topology-control
 class system-cpp-police-dot1x-auth
 class system-cpp-police-protocol-snooping
 class system-cpp-police-forus
 class system-cpp-default

<output truncated>
```

Example: Setting the Default Policer Rates for All CPU Queues

This example shows how to set the policer rates for all CPU queues to their default and then verify the setting.

```
Device> enable
Device# configure terminal
Device(config)# cpp system-default
Defaulting CPP : Policer rate for all classes will be set to their defaults
Device(config)# end
```

Monitoring CoPP

Use these commands to display policer settings, such as, traffic types and policer rates (user-configured and default rates) for CPU queues:

Command	Purpose
show policy-map control-plane	Displays the rates configured for the various traffic types

Command	Purpose
show policy-map system-cpp-policy	Displays all the classes configured under system-cpp policy, and policer rates

CHAPTER 115

Configuring Authorization and Revocation of Certificates in a PKI

This module describes the authorization and revocation of certificates in a public key infrastructure (PKI).

Prerequisites for Authorization and Revocation of Certificates

Plan Your PKI Strategy

Tip It is strongly recommended that you plan your entire PKI strategy before you begin to deploy actual certificates.

Authorization and revocation can occur only after you or a network administrator have completed the following tasks:

- Configured the certificate authority (CA).
- Enrolled peer devices with the CA.
- Identified and configured the protocol (such as IPsec or secure socket layer [SSL]) that is to be used for peer-to-peer communication.

You should decide which authorization and revocation strategy you are going to configure before enrolling peer devices because the peer device certificates might have to contain authorization and revocation-specific information.

High Availability

For high availability, IPsec-secured Stream Control Transmission Protocol (SCTP) must be configured on both the active and the standby devices. For synchronization to work, the redundancy mode on the certificate servers must be set to ACTIVE/STANDBY after you configure SCTP.

Restrictions for Authorization and Revocation of Certificates

- Depending on your Cisco IOS XE release, Lightweight Directory Access Protocol (LDAP) is supported.

Information About Authorization and Revocation of Certificates

PKI Authorization

PKI authentication does not provide authorization. Current solutions for authorization are specific to the router that is being configured, although a centrally managed solution is often required.

There is not a standard mechanism by which certificates are defined as authorized for some tasks and not for others. This authorization information can be captured in the certificate itself if the application is aware of the certificate-based authorization information. But this solution does not provide a simple mechanism for real-time updates to the authorization information and forces each application to be aware of the specific authorization information embedded in the certificate.

When the certificate-based access control list (ACL) mechanism is configured as part of the trustpoint authentication, the application is no longer responsible for determining this authorization information, and it is no longer possible to specify for which application the certificate is authorized. In some cases, the certificate-based ACL on the router gets so large that it cannot be managed. Additionally, it is beneficial to retrieve certificate-based ACL indications from an external server.

Current solutions to the real-time authorization problem involve specifying a new protocol and building a new server (with associated tasks, such as management and data distribution).

PKI and AAA Server Integration for Certificate Status

Integrating your PKI with an authentication, authorization, and accounting (AAA) server provides an alternative online certificate status solution that leverages the existing AAA infrastructure. Certificates can be listed in the AAA database with appropriate levels of authorization. For components that do not explicitly support PKI-AAA, a default label of “all” from the AAA server provides authorization. Likewise, a label of “none” from the AAA database indicates that the specified certificate is not valid. (The absence of any application label is equivalent, but “none” is included for completeness and clarity). If the application component does support PKI-AAA, the component may be specified directly; for example, the application component could be “ipsec,” “ssl,” or “osp.” (ipsec=IP Security, ssl=Secure Sockets Layer, and osp=Open Settlement Protocol.)

Note Currently, no application component supports specification of the application label.

- There may be a time delay when accessing the AAA server. If the AAA server is not available, the authorization fails.

RADIUS or TACACS+ Choosing a AAA Server Protocol

The AAA server can be configured to work with either the RADIUS or TACACS+ protocol. When you are configuring the AAA server for the PKI integration, you must set the RADIUS or TACACS attributes that are required for authorization.

If the RADIUS protocol is used, the password that is configured for the username in the AAA server should be set to "cisco," which is acceptable because the certificate validation provides authentication and the AAA database is only being used for authorization. When the TACACS protocol is used, the password that is configured for the username in the AAA server is irrelevant because TACACS supports authorization without requiring authentication (the password is used for authentication).

In addition, if you are using TACACS, you must add a PKI service to the AAA server. The custom attribute "cert-application=all" is added under the PKI service for the particular user or usergroup to authorize the specific username.

Attribute-Value Pairs for PKI and AAA Server Integration

The table below lists the attribute-value (AV) pairs that are to be used when setting up PKI integration with a AAA server. (Note the values shown in the table are possible values.) The AV pairs must match the client configuration. If they do not match, the peer certificate is not authorized.

Note Users can sometimes have AV pairs that are different from those of every other user. As a result, a unique username is required for each user. The **all** parameter (within the **authorization username** command) specifies that the entire subject name of the certificate will be used as the authorization username.

Table 125: AV Pairs That Must Match

AV Pair	Value
cisco-avpair=pki:cert-application=all	Valid values are "all" and "none."
cisco-avpair=pki:cert-trustpoint=msca	The value is a Cisco IOS XE command-line interface (CLI) configuration trustpoint label. **Note** The cert-trustpoint AV pair is normally optional. If it is specified, the device query must be coming from a certificate trustpoint that has a matching label, and the certificate that is authenticated must have the specified certificate serial number.
cisco-avpair=pki:cert-serial=16318DB7000100001671	The value is a certificate serial number. **Note** The cert-serial AV pair is normally optional. If it is specified, the Cisco device query must be coming from a certificate trustpoint that has a matching label, and the certificate that is authenticated must have the specified certificate serial number.

AV Pair	Value
cisco-avpair=pki:cert-lifetime-end=1:00 jan 1, 2003	The cert-lifetime-end AV pair is available to artificially extend a certificate lifetime beyond the time period that is indicated in the certificate itself. If the cert-lifetime-end AV pair is used, the cert-trustpoint and cert-serial AV pairs must also be specified. The value must match the following form: hours:minutes month day, year. **Note** Only the first three characters of a month are used: Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec. If more than three characters are entered for the month, the remaining characters are ignored (for example Janxxxx).

CRLs or OCSP Server Choosing a Certificate Revocation Mechanism

After a certificate is validated as a properly signed certificate, a certificate revocation method is performed to ensure that the certificate has not been revoked by the issuing CA. Cisco IOS XE software supports two revocation mechanisms--certificate revocation lists (CRLs) and Online Certificate Status Protocol (OCSP). Cisco IOS XE software also supports AAA integration for certificate checking; however, additional authorization functionality is included. For more information on PKI and AAA certificate authorization and status check, see the PKI and AAA Server Integration for Certificate Status section.

The following sections explain how each revocation mechanism works:

What Is a CRL

A certificate revocation list (CRL) is a list of revoked certificates. The CRL is created and digitally signed by the CA that originally issued the certificates. The CRL contains dates for when each certificate was issued and when it expires.

CAs publish new CRLs periodically or when a certificate for which the CA is responsible has been revoked. By default, a new CRL is downloaded after the currently cached CRL expires. An administrator may also configure the duration for which CRLs are cached in router memory or disable CRL caching completely. The CRL caching configuration applies to all CRLs associated with a trustpoint.

When the CRL expires, the router deletes it from its cache. A new CRL is downloaded when a certificate is presented for verification; however, if a newer version of the CRL that lists the certificate under examination is on the server but the router is still using the CRL in its cache, the router does not know that the certificate has been revoked. The certificate passes the revocation check even though it should have been denied.

When a CA issues a certificate, the CA can include in the certificate the CRL distribution point (CDP) for that certificate. Cisco IOS client devices use CDPs to locate and load the correct CRL. The Cisco IOS client supports multiple CDPs, but the Cisco IOS CA currently supports only one CDP; however, third-party vendor CAs may support multiple CDPs or different CDPs per certificate. If a CDP is not specified in the certificate, the client device uses the default Simple Certificate Enrollment Protocol (SCEP) method to retrieve the CRL. (The CDP location can be specified through the **cdp-url**command.)

When implementing CRLs, you should consider the following design considerations:

- CRL lifetimes and the security association (SA) and Internet Key Exchange (IKE) lifetimes.
- The CRL lifetime determines the length of time between CA-issued updates to the CRL. The default CRL lifetime value, which is 168 hours [1 week], can be changed through the **lifetime crl** command.
- The method of the CDP determines how the CRL is retrieved; some possible choices include HTTP, Lightweight Directory Access Protocol (LDAP), SCEP, or TFTP. HTTP, TFTP, and LDAP are the most commonly used methods. Although Cisco IOS software defaults to SCEP, an HTTP CDP is recommended for large installations using CRLs because HTTP can be made highly scalable.
- The location of the CDP determines from where the CRL is retrieved; for example, you can specify the server and file path from which to retrieve the CRL.

Querying All CDPs During Revocation Check

When a CDP server does not respond to a request, the Cisco IOS XE software reports an error, which may result in the peer's certificate being rejected. To prevent a possible certificate rejection and if there are multiple CDPs in a certificate, the Cisco IOS XE software will attempt to use the CDPs in the order in which they appear in the certificate. The device will attempt to retrieve a CRL using each CDP URL or directory specification. If an error occurs using a CDP, an attempt will be made using the next CDP.

Tip Although the Cisco IOS XE software will make every attempt to obtain the CRL from one of the indicated CDPs, it is recommended that you use an HTTP CDP server with high-speed redundant HTTP servers to avoid application timeouts because of slow CDP responses.

What Is OCSP

OCSP is an online mechanism that is used to determine certificate validity and provides the following flexibility as a revocation mechanism:

- OCSP can provide real-time certificate status checking.
- OCSP allows the network administrator to specify a central OCSP server, which can service all devices within a network.
- OCSP also allows the network administrator the flexibility to specify multiple OCSP servers, either per client certificate or per group of client certificates.
- OCSP server validation is usually based on the root CA certificate or a valid subordinate CA certificate, but may also be configured so that external CA certificates or self-signed certificates may be used. Using external CA certificates or self-signed certificates allows the OCSP servers certificate to be issued and validated from an alternative PKI hierarchy.

A network administrator can configure an OCSP server to collect and update CRLs from different CA servers. The devices within the network can rely on the OCSP server to check the certificate status without retrieving and caching each CRL for every peer. When peers have to check the revocation status of a certificate, they send a query to the OCSP server that includes the serial number of the certificate in question and an optional unique identifier for the OCSP request, or a nonce. The OCSP server holds a copy of the CRL to determine if the CA has listed the certificate as being revoked; the server then responds to the peer including the nonce. If the nonce in the response from the OCSP server does not match the original nonce sent by the peer, the response is considered invalid and certificate verification fails. The dialog between the OCSP server and the peer consumes less bandwidth than most CRL downloads.

If the OCSP server is using a CRL, CRL time limitations will be applicable; that is, a CRL that is still valid might be used by the OCSP server although a new CRL has been issued by the CRL containing additional certificate revocation information. Because fewer devices are downloading the CRL information on a regular basis, you can decrease the CRL lifetime value or configure the OCSP server not to cache the CRL. For more information, check your OCSP server documentation.

Note OCSP multiple response handling: Support has been enabled for handling of multiple OCSP single responses from an OCSP responder in a response packet. In addition to the debug log messages the following debug log message will be displayed:

CRYPTO_PKI: Number of single Responses in OCSP response:1(this value can change depending upon the number of responses).

When to Use an OCSP Server

OCSP may be more appropriate than CRLs if your PKI has any of the following characteristics:

- Real-time certificate revocation status is necessary. CRLs are updated only periodically and the latest CRL may not always be cached by the client device. For example, if a client does not yet have the latest CRL cached and a newly revoked certificate is being checked, that revoked certificate will successfully pass the revocation check.
- There are a large number of revoked certificates or multiple CRLs. Caching a large CRL consumes large portions of Cisco IOS memory and may reduce resources available to other processes.
- CRLs expire frequently, causing the CDP to handle a larger load of CRLs.

When to Use Certificate-Based ACLs for Authorization or Revocation

Certificates contain several fields that are used to determine whether a device or user is authorized to perform a specified action.

Because certificate-based ACLs are configured on the device, they do not scale well for large numbers of ACLs; however, certificate-based ACLs do provide very granular control of specific device behavior. Certificate-based ACLs are also leveraged by additional features to help determine when PKI components such as revocation, authorization, or a trustpoint should be used. They provide a general mechanism allowing users to select a specific certificate or a group of certificates that are being validated for either authorization or additional processing.

Certificate-based ACLs specify one or more fields within the certificate and an acceptable value for each specified field. You can specify which fields within a certificate should be checked and which values those fields may or may not have.

There are six logical tests for comparing the field with the value--equal, not equal, contains, does not contain, less than, and greater than or equal. If more than one field is specified within a single certificate-based ACL, the tests of all of the fields within the ACL must succeed to match the ACL. The same field may be specified multiple times within the same ACL. More than one ACL may be specified, and ACL will be processed in turn until a match is found or all of the ACLs have been processed.

Ignore Revocation Checks Using a Certificate-Based ACL

Certificate-based ACLs can be configured to instruct your router to ignore the revocation check and expired certificates of a valid peer. Thus, a certificate that meets the specified criteria can be accepted regardless of the validity period of the certificate, or if the certificate meets the specified criteria, revocation checking does not have to be performed. You can also use a certificate-based ACL to ignore the revocation check when the communication with a AAA server is protected with a certificate.

Ignoring Revocation Lists

To allow a trustpoint to enforce CRLs except for specific certificates, enter the **match certificate**command with the **skip revocation-check** keyword. This type of enforcement is most useful in a hub-and-spoke configuration in which you also want to allow direct spoke-to-spoke connections. In pure hub-and-spoke configurations, all spokes connect only to the hub, so CRL checking is necessary only on the hub. For one spoke to communicate directly with another spoke, the **match certificate**command with the **skip revocation-check** keyword can be used for neighboring peer certificates instead of requiring a CRL on each spoke.

Ignoring Expired Certificates

To configure your router to ignore expired certificates, enter the **match certificate** command with the **allow expired-certificate** keyword. This command has the following purposes:

- If the certificate of a peer has expired, this command may be used to "allow" the expired certificate until the peer can obtain a new certificate.
- If your router clock has not yet been set to the correct time, the certificate of a peer will appear to be not yet valid until the clock is set. This command may be used to allow the certificate of the peer even though your router clock is not set.

Note If Network Time Protocol (NTP) is available only via the IPSec connection (usually via the hub in a hub-and-spoke configuration), the router clock can never be set. The tunnel to the hub cannot be "brought up" because the certificate of the hub is not yet valid.

- "Expired" is a generic term for a certificate that is expired or that is not yet valid. The certificate has a start and end time. An expired certificate, for purposes of the ACL, is one for which the current time of the router is outside the start and end times specified in the certificate.

Skipping the AAA Check of the Certificate

If the communication with an AAA server is protected with a certificate, and you want to skip the AAA check of the certificate, use the **match certificate** command with the **skip authorization-check** keyword. For example, if a virtual private network (VPN) tunnel is configured so that all AAA traffic goes over that tunnel, and the tunnel is protected with a certificate, you can use the **match certificate** command with the **skip authorization-check** keyword to skip the certificate check so that the tunnel can be established.

The **match certificate**command and the **skip authorization-check** keyword should be configured after PKI integration with an AAA server is configured.

Note If the AAA server is available only via an IPSec connection, the AAA server cannot be contacted until after the IPSec connection is established. The IPSec connection cannot be "brought up" because the certificate of the AAA server is not yet valid.

PKI Certificate Chain Validation

A certificate chain establishes a sequence of trusted certificates --from a peer certificate to the root CA certificate. Within a PKI hierarchy, all enrolled peers can validate the certificate of one another if the peers share a trusted root CA certificate or a common subordinate CA. Each CA corresponds to a trustpoint.

When a certificate chain is received from a peer, the default processing of a certificate chain path continues until the first trusted certificate, or trustpoint, is reached. An administrator may configure the level to which a certificate chain is processed on all certificates including subordinate CA certificates.

Configuring the level to which a certificate chain is processed allows for the reauthentication of trusted certificates, the extension of a trusted certificate chain, and the completion of a certificate chain that contains a gap.

Reauthentication of Trusted Certificates

The default behavior is for the device to remove any trusted certificates from the certificate chain sent by the peer before the chain is validated. An administrator may configure certificate chain path processing so that the device does not remove CA certificates that are already trusted before chain validation, so that all certificates in the chain are re-authenticated for the current session.

Extending the Trusted Certificate Chain

The default behavior is for the device to use its trusted certificates to extend the certificate chain if there are any missing certificates in the certificate chain sent by the peer. The device will validate only certificates in the chain sent by the peer. An administrator may configure certificate chain path processing so that the certificates in the peer's certificate chain and the device's trusted certificates are validated to a specified point.

Completing Gaps in a Certificate Chain

An administrator may configure certificate chain processing so that if there is a gap in the configured trustpoint hierarchy, certificates sent by the peer can be used to complete the set of certificates to be validated.

Note If the trustpoint is configured to require parent validation and the peer does not provide the full certificate chain, the gap cannot be completed and the certificate chain is rejected and invalid.

Note It is a configuration error if the trustpoint is configured to require parent validation and there is no parent trustpoint configured. The resulting certificate chain gap cannot be completed and the subordinate CA certificate cannot be validated. The certificate chain is invalid.

How to Configure Authorization and Revocation of Certificates in a PKI

Configuring PKI Integration with a AAA Server

Perform this task to generate a AAA username from the certificate presented by the peer and specify which fields within a certificate should be used to build the AAA database username.

Note The following restrictions should be considered when using the **all** keyword as the subject name for the **authorization username** command:

- Some AAA servers limit the length of the username (for example, to 64 characters). As a result, the entire certificate subject name cannot be longer than the limitation of the server.
- Some AAA servers limit the available character set that may be used for the username (for example, a space [ ] and an equal sign [=] may not be acceptable). You cannot use the **all** keyword for a AAA server having such a character-set limitation.
- The **subject-name** command in the trustpoint configuration may not always be the final AAA subject name. If the fully qualified domain name (FQDN), serial number, or IP address of the router are included in a certificate request, the subject name field of the issued certificate will also have these components. To turn off the components, use the **fqdn**, **serial-number**, and **ip-address** commands with the **none** keyword.
- CA servers sometimes change the requested subject name field when they issue a certificate. For example, CA servers of some vendors switch the relative distinguished names (RDNs) in the requested subject names to the following order: CN, OU, O, L, ST, and C. However, another CA server might append the configured LDAP directory root (for example, O=cisco.com) to the end of the requested subject name.
- Depending on the tools you choose for displaying a certificate, the printed order of the RDNs in the subject name could be different. Cisco IOS software always displays the least significant RDN first, but other software, such as Open Source Secure Socket Layer (OpenSSL), does the opposite. Therefore, if you are configuring a AAA server with a full distinguished name (DN) (subject name) as the corresponding username, ensure that the Cisco IOS software style (that is, with the least significant RDN first) is used.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables the AAA access control model.
Step 4	**aaa authorization network** *listname* [*method*] **Example:** `Device(config)# aaa authorization network maxaaa group tacacs+`	Sets the parameters that restrict user access to a network. • *method* : Can be **group radius**, **group tacacs+**, or **group group-name**.
Step 5	**crypto pki trustpoint** *name* **Example:** `Device(config)# crypto pki trustpoint msca`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.
Step 6	**enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [**pem**] **Example:** `Device(ca-trustpoint)# enrollment url http://caserver.myexample.com` - or- `Device(ca-trustpoint)# enrollment url http://[2001:DB8:1:1::1]:80`	Specifies the following enrollment parameters of the CA: • (Optional) The **mode** keyword specifies the registration authority (RA) mode, if your CA system provides an RA. By default, RA mode is disabled. • (Optional) The **retry period** keyword and *minutes* argument specifies the period, in minutes, in which the router waits before sending the CA another certificate request. Valid values are from 1 to 60. The default is 1. • (Optional) The **retry count** keyword and *number* argument specifies the number of times a router will resend a certificate request when it does not receive a response from the previous request. Valid values are from 1 to 100. The default is 10. • The *url* argument is the URL of the CA to which your router should send certificate requests. **Note** An IPv6 address can be added to the **http:** enrollment method. For example: http://[ipv6-address]:80. The IPv6 address must be enclosed in brackets in the URL.

	Command or Action	Purpose
		• (Optional) The **pem** keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request.
Step 7	revocation-check method **Example:** `Device(ca-trustpoint)# revocation-check crl`	(Optional) Checks the revocation status of a certificate.
Step 8	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 9	**authorization username subjectname** *subjectname* **Example:** `Device(config)# authorization username subjectname serialnumber`	Sets parameters for the different certificate fields that are used to build the AAA username. The *subjectname* argument can be any of the following: • **all**: Entire distinguished name (subject name) of the certificate. • **commonname**: Certification common name. • **country**: Certificate country. • **email**: Certificate e-mail. • **ipaddress**: Certificate IP address. • **locality**: Certificate locality. • **organization**: Certificate organization. • **organizationalunit**: Certificate organizational unit. • **postalcode**: Certificate postal code. • **serialnumber**: Certificate serial number. • **state**: Certificate state field. • **streetaddress**: Certificate street address. • **title**: Certificate title. • **unstructuredname**: Certificate unstructured name.
Step 10	**authorization list** *listname* **Example:**	Specifies the AAA authorization list.

	Command or Action	Purpose
	`Device(config)# authorization list maxaaa`	
Step 11	**tacacs server** *server-name* **Example:** `Device(config)# tacacs server yourserver`	Specifies a TACACS+ server.
Step 12	**address** {**ipv4** | **ipv6**} *ip-address* **Example:** `Device(config-server-tacacs)# address ipv4 192.0.2.2`	Configures the IP address for the TACACS server.
Step 13	**key** *string* **Example:** `Device(config-server-tacacs)# key a_secret_key`	Configures the authorization and encryption key used between the switch and the TACACS server.
Step 14	**end** **Example:** `Device(config-server-tacacs)# end` **Example:**	Returns to privileged EXEC mode.

Troubleshooting Tips

To display debug messages for the trace of interaction (message type) between the CA and the router, use the **debug crypto pki transactions**command. (See the sample output, which shows a successful PKI integration with AAA server exchange and a failed PKI integration with AAA server exchange.)

Successful Exchange

```
Device# debug crypto pki transactions

Apr 22 23:15:03.695: CRYPTO_PKI: Found a issuer match
Apr 22 23:15:03.955: CRYPTO_PKI: cert revocation status unknown.
Apr 22 23:15:03.955: CRYPTO_PKI: Certificate validated without revocation check
```

Each line that shows “CRYPTO_PKI_AAA” indicates the state of the AAA authorization checks. Each of the AAA AV pairs is indicated, and then the results of the authorization check are shown.

```
Apr 22 23:15:04.019: CRYPTO_PKI_AAA: checking AAA authorization (ipsecca_script_aaalist,
PKIAAA-L, <all>)
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: reply attribute ("cert-application" = "all")
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: reply attribute ("cert-trustpoint" = "CA1")
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: reply attribute ("cert-serial" = "15DE")
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: authorization passed
Apr 22 23:12:30.327: CRYPTO_PKI: Found a issuer match
```

Failed Exchange

```
Device# debug crypto pki transactions

Apr 22 23:11:13.703: CRYPTO_PKI_AAA: checking AAA authorization =
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: reply attribute ("cert-application" = "all")
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: reply attribute ("cert-trustpoint"= "CA1")
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: reply attribute ("cert-serial" = "233D")
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: parsed cert-lifetime-end as: 21:30:00
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: timezone specific extended
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: cert-lifetime-end is expired
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: cert-lifetime-end check failed.
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: authorization failed
```

In the above failed exchange, the certificate has expired.

Configuring a Revocation Mechanism for PKI Certificate Status Checking

Perform this task to set up a CRL as the certificate revocation mechanism--CRLs or OCSP--that is used to check the status of certificates in a PKI.

The revocation-check Command

Use the **revocation-check** command to specify at least one method (OCSP, CRL, or skip the revocation check) that is to be used to ensure that the certificate of a peer has not been revoked. For multiple methods, the order in which the methods are applied is determined by the order specified via this command.

If your device does not have the applicable CRL and is unable to obtain one or if the OCSP server returns an error, your device will reject the peer's certificate--unless you include the **none** keyword in your configuration. If the **none** keyword is configured, a revocation check will not be performed and the certificate will always be accepted.

Nonces and Peer Communications with OCSP Servers

When using OCSP, nonces, unique identifiers for OCSP requests, are sent by default during peer communications with your OCSP server. The use of nonces offers a more secure and reliable communication channel between the peer and OCSP server.

If your OCSP server does not support nonces, you may disable the sending of nonces. For more information, see your OCSP server documentation.

Before you begin

- Before issuing any client certificates, the appropriate settings on the server (such as setting the CDP) should be configured.
- When configuring an OCSP server to return the revocation status for a CA server, the OCSP server must be configured with an OCSP response signing certificate that is issued by that CA server. Ensure that the signing certificate is in the correct format, or the router will not accept the OCSP response. See your OCSP manual for additional information.

Note

- OCSP transports messages over HTTP, so there may be a time delay when you access the OCSP server.
- If the OCSP server depends on normal CRL processing to check revocation status, the same time delay that affects CRLs will also apply to OCSP.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Device(config)# crypto pki trustpoint hazel`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.
Step 4	**ocsp url** *url* **Example:** `Device(ca-trustpoint)# ocsp url http://ocsp-server` - or - `Device(ca-trustpoint)# ocsp url http://10.10.10.1:80` - or - `Device(ca-trustpoint)# ocsp url http://[2001DB8:1:1::2]:80`	The *url* argument specifies the URL of an OCSP server so that the trustpoint can check the certificate status. This URL overrides the URL of the OCSP server (if one exists) in the Authority Info Access (AIA) extension of the certificate. All certificates associated with a configured trustpoint are checked by the OCSP server. The URL can be a hostname, IPv4 address, or an IPv6 address.
Step 5	**revocation-check** *method1* [*method2 method3*]] **Example:** `Device(ca-trustpoint)# revocation-check ocsp none`	Checks the revocation status of a certificate. • **crl** —Certificate checking is performed by a CRL. This is the default option. • **none** —Certificate checking is ignored. • **ocsp** —Certificate checking is performed by an OCSP server.

	Command or Action	Purpose
		If a second and third method are specified, each method will be used only if the previous method returns an error, such as a server being down.
Step 6	**ocsp disable-nonce** **Example:** `Device(ca-trustpoint)# ocsp disable-nonce`	(Optional) Specifies that a nonce, or an OCSP request unique identifier, will not be sent during peer communications with the OCSP server.
Step 7	**end** **Example:** `Device(ca-trustpoint)# end`	Exits ca-trustpoint configuration mode and returns to privileged EXEC mode.
Step 8	**show crypto pki certificates** **Example:** `Device# show crypto pki certificates`	(Optional) Displays information about your certificates.
Step 9	**show crypto pki trustpoints** [**status** | *label* [**status**]] **Example:** `Device# show crypto pki trustpoints`	Displays information about the trustpoint configured in router.

Configuring Certificate Authorization and Revocation Settings

Perform this task to specify a certificate-based ACL, to ignore revocation checks or expired certificates, to manually override the default CDP location, to manually override the OCSP server setting, to configure CRL caching, or to set session acceptance or rejection based on a certificate serial number, as appropriate.

Configuring Certificate-Based ACLs to Ignore Revocation Checks

To configure your device to use certificate-based ACLs to ignore revocation checks and expired certificates, perform the following steps:

- Identify an existing trustpoint or create a new trustpoint to be used when verifying the certificate of the peer. Authenticate the trustpoint if it has not already been authenticated. The router may enroll with this trustpoint if you want. Do not set optional CRLs for the trustpoint if you plan to use the **match certificate** command and **skip revocation-check** keyword.
- Determine the unique characteristics of the certificates that should not have their CRL checked and of the expired certificates that should be allowed.
- Define a certificate map to match the characteristics identified in the prior step.
- You can add the **match certificate** command and **skip revocation-check** keyword and the **match certificate command** and **allow expired-certificate** keyword to the trustpoint that was created or identified in the first step.

Note Certificate maps are checked even if the peer's public key is cached. For example, when the public key is cached by the peer, and a certificate map is added to the trustpoint to ban a certificate, the certificate map is effective. This prevents a client with the banned certificate, which was once connected in the past, from reconnecting.

Manually Overriding CDPs in a Certificate

Users can override the CDPs in a certificate with a manually configured CDP. Manually overriding the CDPs in a certificate can be advantageous when a particular server is unavailable for an extended period of time. The certificate's CDPs can be replaced with a URL or directory specification without reissuing all of the certificates that contain the original CDP.

Manually Overriding the OCSP Server Setting in a Certificate

Administrators can override the OCSP server setting specified in the Authority Information Access (AIA) field of the client certificate or set by the issuing the **ocsp url** command. One or more OCSP servers may be manually specified, either per client certificate or per group of client certificates by the **match certificate override ocsp** command. The **match certificate override ocsp**command overrides the client certificate AIA field or the **ocsp url**command setting if a client certificate is successfully matched to a certificate map during the revocation check.

Note Only one OCSP server can be specified per client certificate.

Configuring CRL Cache Control

By default, a new CRL will be downloaded after the currently cached CRL expires. Administrators can either configure the maximum amount of time in minutes a CRL remains in the cache by issuing the **crl cache delete-after** command or disable CRL caching by issuing the **crl cache none** command. Only the **crl-cache delete-after**command or the **crl-cache none** command may be specified. If both commands are entered for a trustpoint, the last command executed will take effect and a message will be displayed.

Neither the **crl-cache none** command nor the **crl-cache delete-after** command affects the currently cached CRL. If you configure the **crl-cache none** command, all CRLs downloaded after this command is issued will not be cached. If you configure the **crl-cache delete-after** command, the configured lifetime will only affect CRLs downloaded after this command is issued.

This functionality is useful is when a CA issues CRLs with no expiration date or with expiration dates days or weeks ahead.

Configuring Certificate Serial Number Session Control

A certificate serial number can be specified to allow a certificate validation request to be accepted or rejected by the trustpoint for a session. A session may be rejected, depending on certificate serial number session control, even if a certificate is still valid. Certificate serial number session control may be configured by using either a certificate map with the **serial-number** field or an AAA attribute, with the **cert-serial-not** command.

Using certificate maps for session control allows an administrator to specify a single certificate serial number. Using the AAA attribute allows an administrator to specify one or more certificate serial numbers for session control.

Before you begin

- The trustpoint should be defined and authenticated before attaching certificate maps to the trustpoint.
- The certificate map must be configured before the CDP override feature can be enabled or the **serial-number** command is issued.
- The PKI and AAA server integration must be successfully completed to use AAA attributes as described in "PKI and AAA Server Integration for Certificate Status."

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki certificate map** *label sequence-number* **Example:** `Device(config)# crypto pki certificate map Group 10`	Defines values in a certificate that should be matched or not matched and enters ca-certificate-map configuration mode.
Step 4	*field-name match-criteria match-value* **Example:** `Device(ca-certificate-map)# subject-name co MyExample`	Specifies one or more certificate fields together with their matching criteria and the value to match. The *field-name* is one of the following case-insensitive name strings or a date: • **alt-subject-name** • **expires-on** • **issuer-name** • **name** • **serial-number** • **subject-name**

	Command or Action	Purpose
		• **unstructured-subject-name** • **valid-start** **Note** Date field format is dd mm yyyy hh:mm:ss or mmm dd yyyy hh:mm:ss. The *match-criteria* is one of the following logical operators: • **co** —contains (valid only for name fields and serial number field) • **eq** —equal (valid for name, serial number, and date fields) • **ge** —greater than or equal (valid only for date fields) • **lt** —less than (valid only for date fields) • **nc** —does not contain (valid only for name fields and serial number field) • **ne** —not equal (valid for name, serial number, and date fields) The *match-value*is the name or date to test with the logical operator assigned by match-criteria. **Note** Use this command only when setting up a certificate-based ACL—not when setting up a certificate-based ACL to ignore revocation checks or expired certificates.
Step 5	**exit** **Example:** `Device(ca-certificate-map)# exit`	Exits ca-certificate-map configuration mode and returns to global configuration mode.
Step 6	**crypto pki trustpoint** *name* **Example:** `Device(config)# crypto pki trustpoint Access2`	Declares the trustpoint, given name and enters ca-trustpoint configuration mode.
Step 7	Do one of the following: • **crl-cache none** • **crl-cache delete-after** *time*	(Optional) Disables CRL caching completely for all CRLs associated with the trustpoint. The **crl-cache none** command does not affect any currently cached CRLs. All CRLs

	Command or Action	Purpose
	Example: `Device(ca-trustpoint)# crl-cache none` **Example:** `Device(ca-trustpoint)# crl-cache delete-after 20`	downloaded after this command is configured will not be cached. (Optional) Specifies the maximum time CRLs will remain in the cache for all CRLs associated with the trustpoint. • *time* —The amount of time in minutes before the CRL is deleted. The **crl-cache delete-after** command does not affect any currently cached CRLs. The configured lifetime will only affect CRLs downloaded after this command is configured.
Step 8	**match certificate** *certificate-map-label* [**allow expired-certificate** | **skip revocation-check** | **skip authorization-check** **Example:** `Device(ca-trustpoint)# match certificate Group skip revocation-check`	(Optional) Associates the certificate-based ACL (that was defined via the **crypto pki certificate map** command) to a trustpoint. • *certificate-map-label* —Must match the *label* argument specified via the **crypto pki certificate map** command. • **allowexpired-certificate** —Ignores expired certificates. • **skip revocation-check** —Allows a trustpoint to enforce CRLs except for specific certificates. • **skip authorization-check** —Skips the AAA check of a certificate when PKI integration with an AAA server is configured.
Step 9	**match certificate** *certificate-map-label* **override cdp** {**url** | **directory**} *string* **Example:** `Device(ca-trustpoint)# match certificate Group1 override cdp url http://server.cisco.com`	(Optional) Manually overrides the existing CDP entries for a certificate with a URL or directory specification. • *certificate-map-label* —A user-specified label that must match the *label* argument specified in a previously defined **crypto pki certificate map** command. • **url** —Specifies that the certificate's CDPs will be overridden with an HTTP or LDAP URL. • **directory** —Specifies that the certificate's CDPs will be overridden with an LDAP directory specification. • *string* —The URL or directory specification.

	Command or Action	Purpose
		Note Some applications may time out before all CDPs have been tried and will report an error message. The error message will not affect the router, and the Cisco IOS software will continue attempting to retrieve a CRL until all CDPs have been tried.
Step 10	**match certificate** *certificate-map-label* **override ocsp** [**trustpoint** *trustpoint-label*] *sequence-number* **url** *ocsp-url* **Example:** `Device(ca-trustpoint)# match certificate mycertmapname override ocsp trustpoint mytp 15 url http://192.0.2.2`	(Optional) Specifies an OCSP server, either per client certificate or per group of client certificates, and may be issued more than once to specify additional OCSP servers and client certificate settings including alternative PKI hierarchies. • *certificate-map-label* —The name of an existing certificate map. • **trustpoint** —The trustpoint to be used when validating the OCSP server certificate. • *sequence-number* —The order the **match certificate override ocsp** command statements apply to the certificate being verified. Matches are performed from the lowest sequence number to the highest sequence number. If more than one command is issued with the same sequence number, it overwrites the previous OCSP server override setting. • **url** —The URL of the OCSP server. When the certificate matches a configured certificate map, the AIA field of the client certificate and any previously issued **ocsp url** command settings are overwritten with the specified OCSP server. If no map-based match occurs, one of the following two cases will continue to apply to the client certificate. • If OCSP is specified as the revocation method, the AIA field value will continue to apply to the client certificate. • If the **ocsp url** configuration exists, the **ocsp url** configuration settings will continue to apply to the client certificates.

	Command or Action	Purpose
Step 11	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits Returns to global configuration mode.
Step 12	**aaa new-model** **Example:** `Device(config)# aaa new-model`	(Optional) Enables the AAA access control model.
Step 13	**aaa attribute list** *list-name* **Example:** `Device(config)# aaa attribute list crl`	(Optional) Defines an AAA attribute list locally on a router and enters config-attr-list configuration mode.
Step 14	**attribute type** {*name*} {*value*} **Example:** `Device(config-attr-list)# attribute type cert-serial-not 6C4A`	(Optional) Defines an AAA attribute type that is to be added to an AAA attribute list locally on a router. To configure certificate serial number session control, an administrator may specify a specific certificate in the *value* field to be accepted or rejected based on its serial number where *name* is set to **cert-serial-not**. If the serial number of the certificate matches the serial number specified by the attribute type setting, the certificate will be rejected. For a full list of available AAA attribute types, execute the **show aaa attributes** command.
Step 15	**exit** **Example:** `Device(ca-trustpoint)# end` **Example:** `Device(config-attr-list)# end`	Returns to privileged EXEC mode.
Step 16	**show crypto pki certificates** **Example:** `Device# show crypto pki certificates`	(Optional) Displays the components of the certificates installed on the router if the CA certificate has been authenticated.

Example

The following is a sample certificate. The OCSP-related extensions are shown using exclamation points.

```
Certificate:
        Data:
            Version: v3
            Serial Number:0x14
            Signature Algorithm:SHAwithRSA - 1.2.840.113549.1.1.4
            Issuer:CN=CA server,OU=PKI,O=Cisco Systems
            Validity:
                Not Before:Thursday, August 8, 2002 4:38:05 PM PST
                Not After:Tuesday, August 7, 2003 4:38:05 PM PST
            Subject:CN=OCSP server,OU=PKI,O=Cisco Systems
            Subject Public Key Info:
                Algorithm:RSA - 1.2.840.113549.1.1.1
                Public Key:
                    Exponent:65537
                    Public Key Modulus:(2048 bits) :
                        <snip>
            Extensions:
                Identifier:Subject Key Identifier - 2.5.29.14
                    Critical:no
                    Key Identifier:
                        <snip>
                Identifier:Authority Key Identifier - 2.5.29.35
                    Critical:no
                    Key Identifier:
                        <snip>
!                Identifier:OCSP NoCheck:- 1.3.6.1.5.5.7.48.1.5
                     Critical:no
                Identifier:Extended Key Usage:- 2.5.29.37
                     Critical:no
                     Extended Key Usage:
                     OCSPSigning
!
                Identifier:CRL Distribution Points - 2.5.29.31
                    Critical:no
                    Number of Points:1
                    Point 0
                        Distribution Point:
[URIName:ldap://CA-server/CN=CA server,OU=PKI,O=Cisco Systems]
        Signature:
            Algorithm:SHAwithRSA - 1.2.840.113549.1.1.4
            Signature:
            <snip>
```

The following example shows an excerpt of the running configuration output when adding a **match certificate override ocsp** command to the beginning of an existing sequence:

```
match certificate map3 override ocsp 5 url http://192.0.2.3/
show running-configuration
.
.
.
        match certificate map3 override ocsp 5 url http://192.0.2.3/
        match certificate map1 override ocsp 10 url http://192.0.2.1/
        match certificate map2 override ocsp 15 url http://192.0.2.2/
```

The following example shows an excerpt of the running configuration output when an existing **match certificate override ocsp** command is replaced and a trustpoint is specified to use an alternative PKI hierarchy:

```
match certificate map4 override ocsp trustpoint tp4 10 url http://192.0.2.4/newvalue
show running-configuration
```

```
.
.
.
        match certificate map3 override ocsp trustpoint tp3 5 url http://192.0.2.3/
        match certificate map1 override ocsp trustpoint tp1 10 url http://192.0.2.1/
        match certificate map4 override ocsp trustpoint tp4 10 url
http://192.0.2.4/newvalue
        match certificate map2 override ocsp trustpoint tp2 15 url http://192.0.2.2/
```

Troubleshooting Tips

If you ignored revocation check or expired certificates, you should carefully check your configuration. Verify that the certificate map properly matches either the certificate or certificates that should be allowed or the AAA checks that should be skipped. In a controlled environment, try modifying the certificate map and determine what is not working as expected.

Configuring Certificate Chain Validation

Perform this task to configure the processing level for the certificate chain path of your peer certificates.

Before you begin

- The device must be enrolled in your PKI hierarchy.
- The appropriate key pair must be associated with the certificate.

Note

- A trustpoint associated with the root CA cannot be configured to be validated to the next level.

The **chain-validation** command is configured with the **continue** keyword for the trustpoint associated with the root CA, an error message will be displayed and the chain validation will revert to the default **chain-validation**command setting.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint***name* **Example:**	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.

	Command or Action	Purpose
	`Device(config)# crypto pki trustpoint ca-sub1`	
Step 4	**chain-validation** [{**stop** | **continue**} [*parent-trustpoint*]] **Example:** `Device(ca-trustpoint)# chain-validation continue ca-sub1`	Configures the level to which a certificate chain is processed on all certificates including subordinate CA certificates. • Use the **stop**keyword to specify that the certificate is already trusted. This is the default setting. • Use the **continue** keyword to specify that the that the subordinate CA certificate associated with the trustpoint must be validated. • The *parent-trustpoint* argument specifies the name of the parent trustpoint the certificate must be validated against.
Step 5	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode

Configuration Examples for Authorization and Revocation of Certificates in a PKI

Configuration and Verification Examples for PKI AAA Authorization

This section provides configuration examples of PKI AAA authorizations:

Example: Device Configuration

The following **show running-config**command output shows the working configuration of a device that is set up to authorize VPN connections using the PKI Integration with AAA Server feature:

```
Device#show running-config

Building configuration...
!
version 17.17
!
hostname catxxxx
!
aaa new-model
!
!
```

```
aaa authentication login default group tacacs+
aaa authentication login no_tacacs enable
aaa authentication dot1x default group tacacs+
aaa authorization exec ACSLab group tacacs+
aaa authorization network ACSLab group tacacs+
aaa accounting exec ACSLab start-stop group tacacs+
aaa accounting network default start-stop group ACSLab
aaa session-id common
!
ip domain name example.com
!
crypto pki trustpoint EM-CERT-SERV
 enrollment url http://192.0.2.33:80
 serial-number
 crl optional
 rsakeypair STOREVPN 2048
 auto-enroll
 authorization list ACSLab
!
crypto pki certificate chain EM-CERT-SERV
 certificate 04
  30820214 3082017D A0030201 02020104 300D0609 2A864886 F70D0101 04050030
  17311530 13060355 0403130C 454D2D43 4552542D 53455256 301E170D 30343031
  31393232 30323535 5A170D30 35303131 38323230 3235355A 3030312E 300E0603
  55040513 07314437 45424434 301C0609 2A864886 F70D0109 02160F37 3230302D
  312E6772 696C2E63 6F6D3081 9F300D06 092A8648 86F70D01 01010500 03818D00
  30818902 818100BD F3B837AA D925F391 2B64DA14 9C2EA031 5A7203C4 92F8D6A8
  7D2357A6 BCC8596F A38A9B10 47435626 D59A8F2A 123195BB BE5A1E74 B1AA5AE0
  5CA162FF 8C3ACA4F B3EE9F27 8B031642 B618AE1B 40F2E3B4 F996BEFE 382C7283
  3792A369 236F8561 8748AA3F BC41F012 B859BD9C DB4F75EE 3CEE2829 704BD68F
  FD904043 0F555702 03010001 A3573055 30250603 551D1F04 1E301C30 1AA018A0
  16861468 7474703A 2F2F3633 2E323437 2E313037 2E393330 0B060355 1D0F0404
  030205A0 301F0603 551D2304 18301680 1420FC4B CF0B1C56 F5BD4C06 0AFD4E67
  341AE612 D1300D06 092A8648 86F70D01 01040500 03818100 79E97018 FB955108
  12F42A56 2A6384BC AC8E22FE F1D6187F DA5D6737 C0E241AC AAAEC75D 3C743F59
  08DEEFF2 0E813A73 D79E0FA9 D62DC20D 8E2798CD 2C1DC3EC 3B2505A1 3897330C
  15A60D5A 8A13F06D 51043D37 E56E45DF A65F43D7 4E836093 9689784D C45FD61D
  EC1F160C 1ABC8D03 49FB11B1 DA0BED6C 463E1090 F34C59E4
  quit
 certificate ca 01
  30820207 30820170 A0030201 02020101 300D0609 2A864886 F70D0101 04050030
  17311530 13060355 0403130C 454D2D43 4552542D 53455256 301E170D 30333132
  31363231 34373432 5A170D30 36313231 35323134 3734325A 30173115 30130603
  55040313 0C454D2D 43455254 2D534552 5630819F 300D0609 2A864886 F70D0101
  01050003 818D0030 81890281 8100C14D 833641CF D784F516 DA6B50C0 7B3CB3C9
  589223AB 99A7DC14 04F74EF2 AAEEE8F5 E3BFAE97 F2F980F7 D889E6A1 2C726C69
  54A29870 7E7363FF 3CD1F991 F5A37CFF 3FFDD3D0 9E486C44 A2E34595 C2D078BB
  E9DE981E B733B868 AA8916C0 A8048607 D34B83C0 64BDC101 161FC103 13C06500
  22D6EE75 7D6CF133 7F1B515F 32830203 010001A3 63306130 0F060355 1D130101
  FF040530 030101FF 300E0603 551D0F01 01FF0404 03020186 301D0603 551D0E04
  16041420 FC4BCF0B 1C56F5BD 4C060AFD 4E67341A E612D130 1F060355 1D230418
  30168014 20FC4BCF 0B1C56F5 BD4C060A FD4E6734 1AE612D1 300D0609 2A864886
  F70D0101 04050003 81810085 D2E386F5 4107116B AD3AC990 CBE84063 5FB2A6B5
  BD572026 528E92ED 02F3A0AE 1803F2AE AA4C0ED2 0F59F18D 7B50264F 30442C41
  0AF19C4E 70BD3CB5 0ADD8DE8 8EF636BD 24410DF4 DB62DAFC 67DA6E58 3879AA3E
  12AFB1C3 2E27CB27 EC74E1FC AEE2F5CF AA80B439 615AA8D5 6D6DEDC3 7F9C2C79
  3963E363 F2989FB9 795BA8
  quit
!
!
crypto isakmp policy 10
 encr aes
 group 14
!
```

```
!
crypto ipsec transform-set ISC_TS_1 esp-aes esp-sha-hmac
!
crypto ipsec profile ISC_IPSEC_PROFILE_2
 set security-association lifetime kilobytes 530000000
 set security-association lifetime seconds 14400
 set transform-set ISC_TS_1
!
!
controller ISA 1/1
!
!
interface Tunnel0
 description MGRE Interface provisioned by ISC
 bandwidth 10000
 ip address 192.0.2.172 255.255.255.0
 no ip redirects
 ip mtu 1408
 ip nhrp map multicast dynamic
 ip nhrp network-id 101
 ip nhrp holdtime 500
 ip nhrp server-only
 no ip split-horizon eigrp 101
 tunnel source GigabitEthernet1/2
 tunnel mode gre multipoint
 tunnel key 101
 tunnel protection ipsec profile ISC_IPSEC_PROFILE_2
!
interface GigabitEthernet1/1
 ip address 192.0.2.1 255.255.255.0
 duplex auto
 speed auto
!
interface GigabitEthernet1/2
 ip address 192.0.2.2 255.255.255.0
 duplex auto
 speed auto
!
!
end
```

Example: Debug of a Successful PKI AAA Authorization

The following **show debugging** command output shows a successful authorization using the PKI Integration with AAA Server feature:

```
Device#show debugging

General OS:
  TACACS access control debugging is on
  AAA Authentication debugging is on
  AAA Authorization debugging is on
Cryptographic Subsystem:
Crypto PKI Trans debugging is on
Device#
May 28 19:36:11.117: CRYPTO_PKI: Trust-Point EM-CERT-SERV picked up
May 28 19:36:12.789: CRYPTO_PKI: Found a issuer match
May 28 19:36:12.805: CRYPTO_PKI: cert revocation status unknown.
May 28 19:36:12.805: CRYPTO_PKI: Certificate validated without revocation check
May 28 19:36:12.813: CRYPTO_PKI_AAA: checking AAA authorization (ACSLab, POD5.example.com,
 <all>)
May 28 19:36:12.813: AAA/BIND(00000042): Bind i/f
May 28 19:36:12.813: AAA/AUTHOR (0x42): Pick method list 'ACSLab'
```

```
May 28 19:36:12.813: TPLUS: Queuing AAA Authorization request 66 for processing
May 28 19:36:12.813: TPLUS: processing authorization request id 66
May 28 19:36:12.813: TPLUS: Protocol set to None .....Skipping
May 28 19:36:12.813: TPLUS: Sending AV service=pki
May 28 19:36:12.813: TPLUS: Authorization request created for 66(POD5.example.com)
May 28 19:36:12.813: TPLUS: Using server 192.0.2.55
May 28 19:36:12.813: TPLUS(00000042)/0/NB_WAIT/203A4628: Started 5 sec timeout
May 28 19:36:12.813: TPLUS(00000042)/0/NB_WAIT: wrote entire 46 bytes request
May 28 19:36:12.813: TPLUS: Would block while reading pak header
May 28 19:36:12.817: TPLUS(00000042)/0/READ: read entire 12 header bytes (expect 27 bytes)
May 28 19:36:12.817: TPLUS(00000042)/0/READ: read entire 39 bytes response
May 28 19:36:12.817: TPLUS(00000042)/0/203A4628: Processing the reply packet
May 28 19:36:12.817: TPLUS: Processed AV cert-application=all
May 28 19:36:12.817: TPLUS: received authorization response for 66: PASS
May 28 19:36:12.817: CRYPTO_PKI_AAA: reply attribute ("cert-application" = "all")
May 28 19:36:12.817: CRYPTO_PKI_AAA: authorization passed
Device#
May 28 19:36:18.681: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 101: Neighbor 192.0.2.171 (Tunnel0) is
 up: new adjacency
Device#
Device# show crypto isakmp sa

dst             src             state          conn-id slot
192.0.2.22     192.0.2.102   QM_IDLE             84    0
```

Example:Debug of a Failed PKI AAA Authorization

The following **show debugging** command output shows that the device is not authorized to connect using VPN. The messages are typical of those that you might see in such a situation.

In this example, the peer username was configured as not authorized, by moving the username to a Cisco Secure ACS group called VPN_Disabled in Cisco Secure ACS. The device, device9.example.com, has been configured to check with a Cisco Secure ACS AAA server prior to establishing a VPN connection to any peer.

```
Device#show debugging

General OS:
  TACACS access control debugging is on
  AAA Authentication debugging is on
  AAA Authorization debugging is on
Cryptographic Subsystem:
  Crypto PKI Trans debugging is on

Device#
May 28 19:48:29.837: CRYPTO_PKI: Trust-Point EM-CERT-SERV picked up
May 28 19:48:31.509: CRYPTO_PKI: Found a issuer match
May 28 19:48:31.525: CRYPTO_PKI: cert revocation status unknown.
May 28 19:48:31.525: CRYPTO_PKI: Certificate validated without revocation check
May 28 19:48:31.533: CRYPTO_PKI_AAA: checking AAA authorization (ACSLab, POD5.example.com,
 <all>)
May 28 19:48:31.533: AAA/BIND(00000044): Bind i/f
May 28 19:48:31.533: AAA/AUTHOR (0x44): Pick method list 'ACSLab'
May 28 19:48:31.533: TPLUS: Queuing AAA Authorization request 68 for processing
May 28 19:48:31.533: TPLUS: processing authorization request id 68
May 28 19:48:31.533: TPLUS: Protocol set to None .....Skipping
May 28 19:48:31.533: TPLUS: Sending AV service=pki
May 28 19:48:31.533: TPLUS: Authorization request created for 68(POD5.example.com)
May 28 19:48:31.533: TPLUS: Using server 192.0.2.55
May 28 19:48:31.533: TPLUS(00000044)/0/NB_WAIT/203A4C50: Started 5 sec timeout
May 28 19:48:31.533: TPLUS(00000044)/0/NB_WAIT: wrote entire 46 bytes request
```

```
May 28 19:48:31.533: TPLUS: Would block while reading pak header
May 28 19:48:31.537: TPLUS(00000044)/0/READ: read entire 12 header bytes (expect 6 bytes)
May 28 19:48:31.537: TPLUS(00000044)/0/READ: read entire 18 bytes response
May 28 19:48:31.537: TPLUS(00000044)/0/203A4C50: Processing the reply packet
May 28 19:48:31.537: TPLUS: received authorization response for 68: FAIL
May 28 19:48:31.537: CRYPTO_PKI_AAA: authorization declined by AAA, or AAA server not found.
May 28 19:48:31.537: CRYPTO_PKI_AAA: No cert-application attribute found. Failing.
May 28 19:48:31.537: CRYPTO_PKI_AAA: authorization failed
May 28 19:48:31.537: CRYPTO_PKI: AAA authorization for list 'ACSLab', and user
'POD5.example.com' failed.
May 28 19:48:31.537: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from 192.0.2.162 is
bad: certificate invalid
May 28 19:48:39.821: CRYPTO_PKI: Trust-Point EM-CERT-SERV picked up
May 28 19:48:41.481: CRYPTO_PKI: Found a issuer match
May 28 19:48:41.501: CRYPTO_PKI: cert revocation status unknown.
May 28 19:48:41.501: CRYPTO_PKI: Certificate validated without revocation check
May 28 19:48:41.505: CRYPTO_PKI_AAA: checking AAA authorization (ACSLab, POD5.example.com,
 <all>)
May 28 19:48:41.505: AAA/BIND(00000045): Bind i/f
May 28 19:48:41.505: AAA/AUTHOR (0x45): Pick method list 'ACSLab'
May 28 19:48:41.505: TPLUS: Queuing AAA Authorization request 69 for processing
May 28 19:48:41.505: TPLUS: processing authorization request id 69
May 28 19:48:41.505: TPLUS: Protocol set to None .....Skipping
May 28 19:48:41.505: TPLUS: Sending AV service=pki
May 28 19:48:41.505: TPLUS: Authorization request created for 69(POD5.example.com)
May 28 19:48:41.505: TPLUS: Using server 198.168.244.55
May 28 19:48:41.509: TPLUS(00000045)/0/IDLE/63B22834: got immediate connect on new 0
May 28 19:48:41.509: TPLUS(00000045)/0/WRITE/63B22834: Started 5 sec timeout
May 28 19:48:41.509: TPLUS(00000045)/0/WRITE: wrote entire 46 bytes request
May 28 19:48:41.509: TPLUS(00000045)/0/READ: read entire 12 header bytes (expect 6 bytes)
May 28 19:48:41.509: TPLUS(00000045)/0/READ: read entire 18 bytes response
May 28 19:48:41.509: TPLUS(00000045)/0/63B22834: Processing the reply packet
May 28 19:48:41.509: TPLUS: received authorization response for 69: FAIL
May 28 19:48:41.509: CRYPTO_PKI_AAA: authorization declined by AAA, or AAA server not found.
May 28 19:48:41.509: CRYPTO_PKI_AAA: No cert-application attribute found. Failing.
May 28 19:48:41.509: CRYPTO_PKI_AAA: authorization failed
May 28 19:48:41.509: CRYPTO_PKI: AAA authorization for list 'ACSLab', and user
'POD5.example.com' failed.
May 28 19:48:41.509: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from 192.0.2.162 is
bad: certificate invalid
Device#
Device# show crypto iskmp sa

dst             src             state          conn-id slot
192.0.2.2        192.0.2.102   MM_KEY_EXCH         95    0
```

Examples: Configuring a Revocation Mechanism

This section contains the following configuration examples that can be used when specifying a revocation mechanism for your PKI:

Example: Configuring an OCSP Server

The following example shows how to configure the router to use the OCSP server that is specified in the AIA extension of the certificate:

```
Device> enable
Device# configure terminal
Device(config)#crypto pki trustpoint mytp
Device(ca-trustpoint)# revocation-check ocsp
```

```
Device(ca-trustpoint)# end
```

Example: Specifying CRL and OCSP Server

The following example shows how to configure the device to download the CRL from the CDP. If the CRL is unavailable, the OCSP server that is specified in the AIA extension of the certificate will be used. If both options fail, certificate verification will also fail.

```
Device> enable
Device# configure terminal
Device(config)#crypto pki trustpoint mytp
Device(ca-trustpoint)#revocation-check crl ocsp
Device(ca-trustpoint)# end
```

Example: Specifying an OCSP Server

The following example shows how to configure your device to use the OCSP server at the HTTP URL "http://myocspserver:81." If the server is down, the revocation check will be ignored.

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint mytp
Device(ca-trustpoint)# ocsp url http://myocspserver:81
Device(ca-trustpoint)# revocation-check ocsp none
Device(ca-trustpoint)# end
```

Example: Disabling Nonces in Communications with OCSP Server

The following example shows communications when a nonce, or a unique identifier for the OCSP request, is disabled for communications with the OCSP server:

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint mytp
Device(ca-trustpoint)# ocsp url http://myocspserver:81
Device(ca-trustpoint)# revocation-check ocsp none
Device(ca-trustpoint)# ocsp disable-nonce
Device(ca-trustpoint)# end
```

Example: Configuring a Hub Device for Certificate Revocation Checks

The following example shows a hub device at a central site that is providing connectivity for several branch offices to the central site.

The branch offices are also able to communicate directly with each other using additional IPSec tunnels between the branch offices.

The CA publishes CRLs on an HTTP server at the central site. The central site checks CRLs for each peer when setting up an IPSec tunnel with that peer.

The example does not show the IPSec configuration--only the PKI-related configuration is shown.

Home Office Hub Configuration

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint VPN-GW
Device(ca-trustpoint)# enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
Device(ca-trustpoint)# serial-number none
Device(ca-trustpoint)# fqdn none
Device(ca-trustpoint)# ip-address none
Device(ca-trustpoint)# subject-name o=Home Office Inc,cn=Central VPN Gateway
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# end
```

Central Site Hub Device

```
Device# show crypto ca certificate

Certificate
  Status: Available
  Certificate Serial Number: 2F62BE14000000000CA0
  Certificate Usage: General Purpose
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    Name: Central VPN Gateway
    cn=Central VPN Gateway
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 00:43:26 GMT Sep 26 2003
    end   date: 00:53:26 GMT Sep 26 2004
    renew date: 00:00:00 GMT Jan 1 1970
  Associated Trustpoints: VPN-GW
CA Certificate
  Status: Available
  Certificate Serial Number: 1244325DE0369880465F977A18F61CA8
  Certificate Usage: Signature
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    cn=Central Certificate Authority
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 22:19:29 GMT Oct 31 2002
    end   date: 22:27:27 GMT Oct 31 2017
  Associated Trustpoints: VPN-GW
```

Trustpoint on the Branch Office Device

```
Device> enable
Device# configure terminal
Device(ca-trustpoint)# crypto pki trustpoint home-office
Device(ca-trustpoint)# enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
Device(ca-trustpoint)# serial-number none
Device(ca-trustpoint)# fqdn none
```

```
Device(ca-trustpoint)# ip-address none
Device(ca-trustpoint)# subject-name o=Home Office Inc,cn=Branch 1
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# end
```

A certificate map is entered on the branch office device.

```
branch1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
branch1(config)# crypto pki certificate map central-site 10
branch1(ca-certificate-map)# end
```

The output from the **show certificate** command on the central site hub device shows that the certificate was issued by the following:

```
cn=Central Certificate Authority
o=Home Office Inc
```

These two lines are combined into one line using a comma (,) to separate them, and the original lines are added as the first criteria for a match.

```
Device(ca-certificate-map)# issuer-name co cn=Central Certificate Authority, ou=Home Office
 Inc
!The above line wrapped but should be shown on one line with the line above it.
```

The same combination is done for the subject name from the certificate on the central site device (note that the line that begins with “Name:” is not part of the subject name and must be ignored when creating the certificate map criteria). This is the subject name to be used in the certificate map.

cn=Central VPN Gateway

o=Home Office Inc

```
Device(ca-certificate-map)# subject-name eq cn=central vpn gateway, o=home office inc
```

Now the certificate map is added to the trustpoint that was configured earlier.

```
Device> enable
Device# configure terminal
Device(ca-certificate-map)# crypto pki trustpoint home-office
Device(ca-trustpoint)# match certificate central-site skip revocation-check
Device(ca-trustpoint)# end
```

The configuration is checked (most of configuration is not shown).

```
Device# write term

!Many lines left out
.
.
.
crypto pki trustpoint home-office
 enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
 serial-number none
 fqdn none
 ip-address none
 subject-name o=Home Office Inc,cn=Branch 1
 revocation-check crl
```

```
 match certificate central-site skip revocation-check
!
!
crypto pki certificate map central-site 10
 issuer-name co cn = Central Certificate Authority, ou = Home Office Inc
 subject-name eq cn = central vpn gateway, o = home office inc
!many lines left out
```

Note that the issuer-name and subject-name lines have been reformatted to make them consistent for later matching with the certificate of the peer.

If the branch office is checking the AAA, the trustpoint will have lines similar to the following:

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint home-office
Device(ca-trustpoint)# authorization list allow_list
Device(ca-trustpoint)# authorization username subjectname commonname
Device(ca-trustpoint)# end
```

After the certificate map has been defined as was done above, the following command is added to the trustpoint to skip AAA checking for the central site hub.

```
Device(ca-trustpoint)# match certificate central-site skip authorization-check
```

In both cases, the branch site device has to establish an IPSec tunnel to the central site to check CRLs or to contact the AAA server. However, without the **match certificate**command and **central-site skip authorization-check (argument and keyword)**, the branch office cannot establish the tunnel until it has checked the CRL or the AAA server. (The tunnel will not be established unless the **match certificate**command and **central-site skip authorization-check** argument and keyword are used.)

The **match certificate** command and **allow expired-certificate** keyword would be used at the central site if the device at a branch site had an expired certificate and it had to establish a tunnel to the central site to renew its certificate.

Trustpoint on the Central Site Device

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint VPN-GW
Device(ca-trustpoint)# enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
Device(ca-trustpoint)# serial-number none
Device(ca-trustpoint)# fqdn none
Device(ca-trustpoint)# ip-address none
Device(ca-trustpoint)# subject-name o=Home Office Inc,cn=Central VPN Gateway
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# end
```

Trustpoint on the Branch 1 Site Device

```
Device# show crypto ca certificate

Certificate
  Status: Available
  Certificate Serial Number: 2F62BE14000000000CA0
  Certificate Usage: General Purpose
  Issuer:
    cn=Central Certificate Authority
```

```
    o=Home Office Inc
  Subject:
    Name: Branch 1 Site
    cn=Branch 1 Site
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 00:43:26 GMT Sep 26 2003
    end   date: 00:53:26 GMT Oct 3 2003
    renew date: 00:00:00 GMT Jan 1 1970
  Associated Trustpoints: home-office
CA Certificate
  Status: Available
  Certificate Serial Number: 1244325DE0369880465F977A18F61CA8
  Certificate Usage: Signature
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    cn=Central Certificate Authority
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 22:19:29 GMT Oct 31 2002
    end   date: 22:27:27 GMT Oct 31 2017
  Associated Trustpoints: home-office
```

A certificate map is entered on the central site device.

```
Device> enable
Device# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)# crypto pki certificate map branch1 10
Device(ca-certificate-map)# issuer-name co cn=Central Certificate Authority, ou=Home Office
 Inc
!The above line wrapped but should be part of the line above it.
Device(ca-certificate-map)# subject-name eq cn=Brahcn 1 Site,o=home office inc
Device(ca-certificate-map)# end
```

The certificate map is added to the trustpoint.

```
Device> enable
Device# configure terminal
Device(ca-certificate-map)# crypto pki trustpoint VPN-GW
Device(ca-trustpoint)# match certificate branch1 allow expired-certificate
Device(ca-trustpoint)# exit
Device (config) #exit
```

The configuration should be checked (most of the configuration is not shown).

```
Device# write term

!many lines left out
crypto pki trustpoint VPN-GW
 enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
 serial-number none
 fqdn none
 ip-address none
 subject-name o=Home Office Inc,cn=Central VPN Gateway
 revocation-check crl
 match certificate branch1 allow expired-certificate
!
```

```
!
crypto pki certificate map central-site 10
 issuer-name co cn = Central Certificate Authority, ou = Home Office Inc
 subject-name eq cn = central vpn gateway, o = home office inc
! many lines left out
```

The **match certificate**command and **branch1 allow expired-certificate** (argument and keyword) and the certificate map should be removed as soon as the branch device has a new certificate.

Examples: Configuring Certificate Authorization and Revocation Settings

This section contains the following configuration examples that can be used when specifying a CRL cache control setting or certificate serial number session control:

Example: Configuring CRL Cache Control

The following example shows how to disable CRL caching for all CRLs associated with the CA1 trustpoint:

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint CA1
Device(ca-trustpoint)# enrollment url http://CA1:80
Device(ca-trustpoint)# ip-address gigabitethernet1/1
Device(ca-trustpoint)# crl query ldap://ldap_CA1
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# crl cache none
Device(ca-trustpoint)# end
```

The current CRL is still cached immediately after executing the example configuration shown above:

Device# **show crypto pki crls**

```
CRL Issuer Name:
    cn=name Cert Manager,ou=pki,o=example.com,c=US
    LastUpdate: 18:57:42 GMT Nov 26 2005
    NextUpdate: 22:57:42 GMT Nov 26 2005
    Retrieved from CRL Distribution Point:
      ldap://ldap.example.com/CN=name Cert Manager,O=example.com
```

When the current CRL expires, a new CRL is then downloaded to the router at the next update. The **crl-cache none**command takes effect and all CRLs for the trustpoint are no longer cached; caching is disabled. You can verify that no CRL is cached by executing the **show crypto pki crls** command. No output will be shown because there are no CRLs cached.

The following example shows how to configure the maximum lifetime of 2 minutes for all CRLs associated with the CA1 trustpoint:

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint CA1
Device(ca-trustpoint)# enrollment url http://CA1:80
Device(ca-trustpoint)# ip-address gigabitethernet1/1
Device(ca-trustpoint)# crl query ldap://ldap_CA1
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# crl cache delete-after 2
Device(ca-trustpoint)# end
```

The current CRL is still cached immediately after executing the example configuration above for setting the maximum lifetime of a CRL:

Device# **show crypto pki crls**

```
CRL Issuer Name:
    cn=name Cert Manager,ou=pki,o=example.com,c=US
    LastUpdate: 18:57:42 GMT Nov 26 2005
    NextUpdate: 22:57:42 GMT Nov 26 2005
    Retrieved from CRL Distribution Point:
      ldap://ldap.example.com/CN=name Cert Manager,O=example.com
When the current CRL expires, a new CRL is downloaded to the router at the next update and
 the crl-cache delete-after
command takes effect. This newly cached CRL and all subsequent CRLs will be deleted after
a maximum lifetime of 2 minutes.
You can verify that the CRL will be cached for 2 minutes by executing the show crypto pki
crls
 command. Note that the NextUpdate time is 2 minutes after the LastUpdate time.
```

Device# **show crypto pki crls**

```
CRL Issuer Name:
    cn=name Cert Manager,ou=pki,o=example.com,c=US
    LastUpdate: 22:57:42 GMT Nov 26 2005

    NextUpdate: 22:59:42 GMT Nov 26 2005
    Retrieved from CRL Distribution Point:
```

ldap://ldap.example.com/CN=name Cert Manager,O=example.com

Example: Configuring Certificate Serial Number Session Control

The following example shows the configuration of certificate serial number session control using a certificate map for the CA1 trustpoint:

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint CA1
Device(ca-trustpoint)# enrollment url http://CA1
Device(ca-trustpoint)# chain-validation stop
Device(ca-trustpoint)# crl query ldap://ldap_server
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# match certificate crl
Device(ca-trustpoint)# exit
Device(config)# crypto pki certificate map crl 10
Device(ca-certificate-map)# serial-number co 279d
Device(ca-certificate-map)# end
```

Note If the *match-criteria* value is set to **eq** (equal) instead of **co** (contains), the serial number must match the certificate map serial number exactly, including any spaces.

The following example shows the configuration of certificate serial number session control using AAA attributes. In this case, all valid certificates will be accepted if the certificate does not have the serial number "4ACA."

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint CA1
Device(ca-trustpoint)# enrollment url http://CA1
Device(ca-trustpoint)# ip-address GigabitEthernet1/1
```

```
Device(ca-trustpoint)# crl query ldap://ldap_CA1
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# exit
Device(config)# aaa new-model
Device(config)# aaa attribute list crl
Device(config-attr-list)# attribute-type aaa-cert-serial-not 4ACA
Device(config-attr-list)# end
```

The server log shows that the certificate with the serial number "4ACA" was rejected. The certificate rejection is shown using exclamation points.

```
.
.
.
Dec 3 04:24:39.051: CRYPTO_PKI: Trust-Point CA1 picked up
Dec 3 04:24:39.051: CRYPTO_PKI: locked trustpoint CA1, refcount is 1
Dec 3 04:24:39.051: CRYPTO_PKI: unlocked trustpoint CA1, refcount is 0
Dec 3 04:24:39.051: CRYPTO_PKI: locked trustpoint CA1, refcount is 1
Dec 3 04:24:39.135: CRYPTO_PKI: validation path has 1 certs
Dec 3 04:24:39.135: CRYPTO_PKI: Found a issuer match
Dec 3 04:24:39.135: CRYPTO_PKI: Using CA1 to validate certificate
Dec 3 04:24:39.135: CRYPTO_PKI: Certificate validated without revocation check
Dec 3 04:24:39.135: CRYPTO_PKI: Selected AAA username: 'PKIAAA'
Dec 3 04:24:39.135: CRYPTO_PKI: Anticipate checking AAA list:'CRL'
Dec 3 04:24:39.135: CRYPTO_PKI_AAA: checking AAA authorization (CRL, PKIAAA-L1, <all>)
Dec 3 04:24:39.135: CRYPTO_PKI_AAA: pre-authorization chain validation status (0x4)
Dec 3 04:24:39.135: AAA/BIND(00000021): Bind i/f
Dec 3 04:24:39.135: AAA/AUTHOR (0x21): Pick method list 'CRL'
.
.
.
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: reply attribute ("cert-application" = "all")
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: reply attribute ("cert-trustpoint" = "CA1")
!
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: reply attribute ("cert-serial-not" = "4ACA")
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: cert-serial doesn't match ("4ACA" != "4ACA")
!
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: post-authorization chain validation status (0x7)
!
Dec 3 04:24:39.175: CRYPTO_PKI: AAA authorization for list 'CRL', and user 'PKIAAA' failed.
Dec 3 04:24:39.175: CRYPTO_PKI: chain cert was anchored to trustpoint CA1, and chain
validation result was:
 CRYPTO_PKI_CERT_NOT_AUTHORIZED
!
Dec 3 04:24:39.175: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from 192.0.2.43 is bad:
 certificate invalid
Dec 3 04:24:39.175: %CRYPTO-6-IKMP_MODE_FAILURE: Processing of Main mode failed with peer
at 192.0.2.43
.
.
.
```

Examples: Configuring Certificate Chain Validation

This section contains the following configuration examples that can be used to specify the level of certificate chain processing for your device certificates:

Configuring Certificate Chain Validation from Peer to RootCA

In the following configuration example, all of the certificates will be validated--the peer, SubCA11, SubCA1, and RootCA certificates.

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint RootCA
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation stop
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair RootCA
Device(ca-trustpoint)# exit
Device(config)# crypto pki trustpoint SubCA1
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation continue RootCA
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair SubCA1
Device(ca-trustpoint)# exit
Device(config)# crypto pki trustpoint SubCA11
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation continue SubCA1
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair SubCA11
Device(ca-trustpoint)# end
```

Configuring Certificate Chain Validation from Peer to Subordinate CA

In the following configuration example, the following certificates will be validated--the peer and SubCA1 certificates.

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint RootCA
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation stop
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair RootCA
Device(ca-trustpoint)# exit
Device(config)# crypto pki trustpoint SubCA1
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation continue RootCA
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair SubCA1
Device(ca-trustpoint)# exit
Device(config)# crypto pki trustpoint SubCA11
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation continue SubCA1
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair SubCA11
Device(ca-trustpoint)# end
```

Configuring Certificate Chain Validation Through a Gap

In the following configuration example, SubCA1 is not in the configured Cisco IOS hierarchy but is expected to have been supplied in the certificate chain presented by the peer.

If the peer supplies the SubCA1 certificate in the presented certificate chain, the following certificates will be validated--the peer, SubCA11, and SubCA1 certificates.

If the peer does not supply the SubCA1 certificate in the presented certificate chain, the chain validation will fail.

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint RootCA
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation stop
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair RootCA
Device(ca-trustpoint)# exit
Device(config)# crypto pki trustpoint SubCA11
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation continue RootCA
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair SubCA11
Device(ca-trustpoint)# end
```

CHAPTER 116

Secure Operation in FIPS Mode

FIPS 140-2 Overview

The Federal Information Processing Standards (FIPS) Publication 140-2 (Security Requirements for Cryptographic Modules) details the U.S and Canadian governments' requirements for cryptographic modules. FIPS 140-2 specifies certain cryptographic algorithms as secure, and it also identifies which algorithms should be used if a cryptographic module is to be called FIPS compliant. For more information on the FIPS 140-2 standard and validation program, refer National Institute of Standards and Technology (NIST) website.

The FIPS 140-2 Compliance Review (CR) documents for switches are posted on the following website:

https://www.cisco.com/c/en/us/solutions/industries/government/global-government-certifications/fips-140.html

Click the link in the **Certification Date** column to view the CR Certificate.

Security Policy document describes the FIPS implementation, hardware installation, firmware initialization, and software configuration procedures for FIPS operation. You can access the FIPS 140-2 Consolidated Validation Certificate and Security Policy document on NIST Computer Security Resource Center. This website opens a Search window. In the **Vendor** field, enter "Cisco" and click **Search**. The resulting window provides a list of Cisco platforms that are FIPS Compliant. From the list, click the desired platform to obtain its Security Policy and Consolidated Certificate.

Important This document describes FIPS mode behavior for switches in general. For more information on platform-specific FIPS 140-2 implementation, refer the FIPS 14-2 Security Policy document for the platform.

Configure FIPS 140-2

Following is a generic procedure to enable FIPS mode of operation for switches. For a detailed configuration procedure, refer FIPS 140-2 Security Policy document for the required device.

Procedure

Step 1 (Optional) Enable FIPS 140-2 logging.

Example:

```
Device(config)# logging console errors
```

Step 2 Configure Authorization key.

Example:

```
Device(config)# fips authorization-key key
```

Note that *key* is 128 bits, which is, 16 HEX byte key.

What to do next

After you enable FIPS, reboot the system to start operating in FIPS mode.

Key Zeroization

A critical FIPS requirement is the capability to zeroize keys and passwords in the event of unsafe state triggers during FIPS mode of operation.

You can delete the FIPS authorization keys using the **no fips authorization-key** command in global configuration mode. This command deletes the key from flash. A reboot takes the system out of FIPS mode of operation.

If there is a security breach, use the **fips zeroize** command to delete all data including the running configuration, Trust Anchor Module, FIPS authorization keys, all ISE Server certificates, and IOS image in flash.

The system reboots after this command is executed.

Caution FIPS zeroization is a critical step where all data is lost. Use it with caution.

Session keys are zeroized by the protocols programmatically.

```
Device(config)#fips zeroize

**Critical Warning** - This command is irreversible
and will zeroize the FVPK by Deleting the IOS
image and config files, please use extreme
caution and confirm with Yes on each of three
iterations to complete. The system will reboot
after the command executes successfully
Proceed ?? (yes/[no]):
```

Disable FIPS Mode

You can disable FIPS mode using the **no fips authorization-key** command.

The **no fips authorization-key** command deletes the authorization key from flash. Note that the authorization key is operational until you reload the switch.

To completely remove the authorization key and disable FIPS mode, reload the switch.

```
Device> enable
Device# config terminal
Device(config)# no fips authorization-key
Device(config)# end
```

Verify FIPS Configuration

Use the **show fips status** command to display the FIPS configuration information.

Use the **show fips authorization-key** command to display the hashed FIPS key.

Note FIPS configuration information does not appear when you list the active configuration using the **show running-config** command or when you list the startup configuration using the **show startup-config** command.

The following are sample outputs of the **show** commands:

```
Device# show fips authorization-key

FIPS: Stored key (16) : 11111111111111111111111111111111
```

```
Device#show romvar

ROMMON variables:
PS1="switch: "
BOARDID="24666"
SWITCH_NUMBER="1"
TERMLINES="0"
MOTHERBOARD_ASSEMBLY_NUM="73-18506-02"
MOTHERBOARD_REVISION_NUM="04"
MODEL_REVISION_NUM="P2A"
POE1_ASSEMBLY_NUM="73-16123-03"
POE1_REVISION_NUM="A0"
POE1_SERIAL_NUM="FOC21335EF2"
POE2_ASSEMBLY_NUM="73-16123-03"
POE2_REVISION_NUM="A0"
POE2_SERIAL_NUM="FOC21335EF3"
IMAGE_UPGRADE="no"
MAC_ADDR="F8:7B:20:77:F7:80"
MODEL_NUM="XXXXX-XXXX"
MOTHERBOARD_SERIAL_NUM="FOC21351BC3"
BAUD="9600"
SYSTEM_SERIAL_NUM="FCW2138L0AF"
USB_SERIAL_NUM="FOC213609Y5"
STKPWR_SERIAL_NUM="FOC21360HTS"
```

```
STKPWR_ASSEMBLY_NUM="73-11956-08"
STKPWR_REVISION_NUM="B0"
USB_ASSEMBLY_NUM="73-16167-02"
USB_REVISION_NUM="A0"
TAN_NUM="68-101202-01"
TAN_REVISION_NUMBER="23"
VERSION_ID="P2A"
CLEI_CODE_NUMBER="ABCDEFGHIJ"
ECI_CODE_NUMBER="123456"
TAG_ID="E20034120133FC00062B0965"
IP_SUBNET_MASK="255.255.0.0"
TEMPLATE="access"
TFTP_BLKSIZE="8192"
ENABLE_BREAK="yes"
TFTP_SERVER="10.8.0.6"
DEFAULT_GATEWAY="10.8.0.1"
IP_ADDRESS="10.8.3.33"
CRASHINFO="crashinfo:crashinfo_RP_00_00_20180420-020851-PDT"
CALL_HOME_DEBUG="0000000000000"
IP_ADDR="172.21.226.35/255.255.255.0"
DEFAULT_ROUTER="10.5.49.254"
RET_2_RTS=""
FIPS_KEY="5AC9BCA165E85D9FA3F2E5FC96AD98E8F943FBAB79B93E78"
MCP_STARTUP_TRACEFLAGS="00000000:00000000"
AUTOREBOOT_RESTORE="0"
MANUAL_BOOT="yes"
<output truncated>
Device#
```

CHAPTER 117

Cisco TrustSec Overview

Cisco TrustSec builds secure networks by establishing domains of trusted network devices. Each device in the domain is authenticated by its peers. Communication on the links between devices in the domain is secured with a combination of encryption, message integrity check, and data-path replay protection mechanisms.

Restrictions for Cisco TrustSec

- Protected access credential (PAC) provisioning fails and remains in hung state, when an invalid device ID is specified. Even after clearing the PAC, and configuring the correct device ID and password, PAC still fails.

  As a workaround, in the Cisco Identity Services Engine (ISE), uncheck the Suppress Anomalous Clients option in the Administration> System> Settings> Protocols> Radius menu for PAC to work.

- Cisco TrustSec is not supported in FIPS mode when PAC is enabled.

- Cisco TrustSec over Radsec is not supported.

Restrictions for configuring Cisco TrustSec in PAC-less mode:

- PAC-less mode is only supported on ISE 3.4.x and later.

- When the device is in PAC-less mode, all servers within the server group must be configured with the PAC-Less configuration (key). Mixing configurations, such as having one server with a PAC key configuration and another with PAC-less configuration, is not allowed.

- In PAC-less mode, the Cisco TrustSec credential command with the device ID is the only parameterneeded to download environment data. However, SGACL requests do not require any credential information.
- Device in PAC-Less mode can be identified by “cts-pac-less” attribute by radius debug.
- IPv6 support for PAC-less is not available.
- Multi-ISE support is limited to up to 2 ISEs.

Information About Cisco TrustSec Architecture

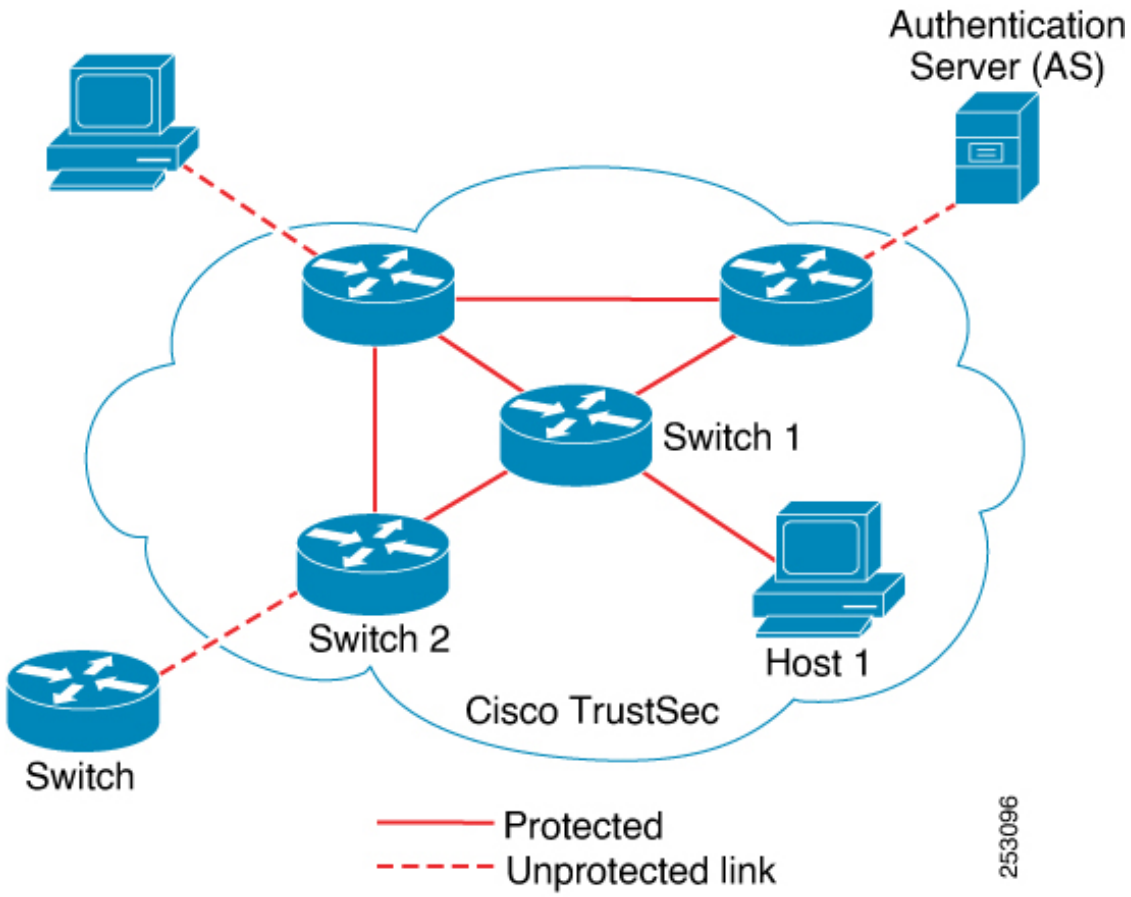
The Cisco TrustSec security architecture builds secure networks by establishing domains of trusted network devices. Each device in the domain is authenticated by its peers. Communication on the links between devices in the domain is secured with a combination of encryption, message integrity check, and data-path replay protection mechanisms. Cisco TrustSec uses the device and user credentials acquired during authentication for classifying the packets by security groups (SGs) as they enter the network. This packet classification is maintained by tagging packets on ingress to the Cisco TrustSec network so that they can be properly identified for the purpose of applying security and other policy criteria along the data path. The tag, called the security group tag (SGT), allows the network to enforce the access control policy by enabling the endpoint device to act upon the SGT to filter traffic.

The Cisco TrustSec architecture incorporates three key components:

- Authenticated networking infrastructure—After the first device (called the seed device) authenticates with the authentication server to begin the Cisco TrustSec domain, each new device added to the domain is authenticated by its peer devices already within the domain. The peers act as intermediaries for the domain’s authentication server. Each newly-authenticated device is categorized by the authentication server and assigned a security group number based on its identity, role, and security posture.
- Security group-based access control—Access policies within the Cisco TrustSec domain are topology-independent, based on the roles (as indicated by security group number) of source and destination devices rather than on network addresses. Individual packets are tagged with the security group number of the source.
- Secure communication—With encryption-capable hardware, communication on each link between devices in the domain can be secured with a combination of encryption, message integrity checks, and data-path replay protection mechanisms.

The following figure shows an example of a Cisco TrustSec domain. In this example, several networking devices and an endpoint device are inside the Cisco TrustSec domain. One endpoint device and one networking device are outside the domain because they are not Cisco TrustSec-capable devices or because they have been refused access. The authentication server is considered to be outside of the Cisco TrustSec domain; it is either a Cisco Identities Service Engine (Cisco ISE), or a Cisco Secure Access Control System (Cisco ACS).

Figure 121: Cisco TrustSec Network Domain Example



Each participant in the Cisco TrustSec authentication process acts in one of the following roles:

- Supplicant—An unauthenticated device connected to a peer within the Cisco TrustSec domain, and attempting to join the Cisco TrustSec domain.
- Authentication server—The server that validates the identity of the supplicant and issues the policies that determine the supplicant's access to services within the Cisco TrustSec domain.
- Authenticator—An authenticated device that is already part of the Cisco TrustSec domain and can authenticate new peer supplicants on behalf of the authentication server.

When the link between a supplicant and an authenticator first comes up, the following sequence of events typically occurs:

1. Authentication (802.1X)—The supplicant is authenticated by the authentication server, with the authenticator acting as an intermediary. Mutual authentication is performed between the two peers (supplicant and authenticator).
2. Authorization—Based on the identity information of the supplicant, the authentication server provides authorization policies, such as security group assignments and ACLs, to each of the linked peers. The authentication server provides the identity of each peer to the other, and each peer then applies the appropriate policy for the link.

3. Security Association Protocol (SAP) negotiation—When both sides of a link support encryption, the supplicant and the authenticator negotiate the necessary parameters to establish a security association (SA).

When all three steps are complete, the authenticator changes the state of the link from the unauthorized (blocking) state to the authorized state, and the supplicant becomes a member of the Cisco TrustSec domain.

Cisco TrustSec uses ingress tagging and egress filtering to enforce access control policy in a scalable manner. Packets entering the domain are tagged with a security group tag (SGT) containing the assigned security group number of the source device. This packet classification is maintained along the data path within the Cisco TrustSec domain for the purpose of applying security and other policy criteria. The final Cisco TrustSec device on the data path, either the endpoint or network egress point, enforces an access control policy based on the security group of the Cisco TrustSec source device and the security group of the final Cisco TrustSec device. Unlike traditional access control lists based on network addresses, Cisco TrustSec access control policies are a form of role-based access control lists (RBACLs) called security group access control lists (SGACLs).

Note Ingress refers to packets entering the first Cisco TrustSec-capable device encountered by a packet on its path to the destination and egress refers to packets leaving the last Cisco TrustSec-capable device on the path.

Security Group-Based Access Control

This section provides information about security group-based access control lists (SGACLs).

Security Groups and SGTs

A security group is a grouping of users, endpoint devices, and resources that share access control policies. Security groups are defined by the administrator in the Cisco ISE or Cisco Secure ACS. As new users and devices are added to the Cisco TrustSec domain, the authentication server assigns these new entities to appropriate security groups. Cisco TrustSec assigns to each security group a unique 16-bit security group number whose scope is global within a Cisco TrustSec domain. The number of security groups in the device is limited to the number of authenticated network entities. You do not have to manually configure security group numbers.

Once a device is authenticated, Cisco TrustSec tags any packet that originates from that device with a security group tag (SGT) that contains the security group number of the device. The packet carries this SGT throughout the network within the Cisco TrustSec header. The SGT is a single label that determines the privileges of the source within the entire enterprise.

Because the SGT contains the security group of the source, the tag can be referred to as the source SGT. The destination device is also assigned to a security group (the destination SG) that can be referred to for simplicity as the destination group tag (DGT), although the actual Cisco TrustSec packet tag does not contain the security group number of the destination device.

Security Group ACL Support

Security group access control lists (SGACLs) is a policy enforcement through which the administrator can control operations performed by an user, based on security group assignments and destination resources. Policy enforcement within the Cisco Trustsec domain is represented by a permissions matrix, with source

security group number on one axis and destination security group number on the other axis. Each cell in the matrix contains an ordered list of SGACLs, which specifies permissions that should be applied to packets originating from an IP belonging to a source security group and having a destination IP that belongs to the destination security group.

SGACL provides stateless access control mechanism based on the security association or security group tag value instead of IP addresses and filters. There are three ways to provision an SGACL policy:

- Static policy provisioning: The SGACL policies are defined by the user using the command **cts role-based permission**.
- Dynamic policy provisioning: Configuration of SGACL policies should be done primarily through the policy management function of the Cisco Secure ACS or the Cisco Identity Services Engine.
- Change of Authorization (CoA): The updated policy is downloaded when the SGACL policy is modified on the ISE and CoA is pushed to the Cisco TrustSec device.

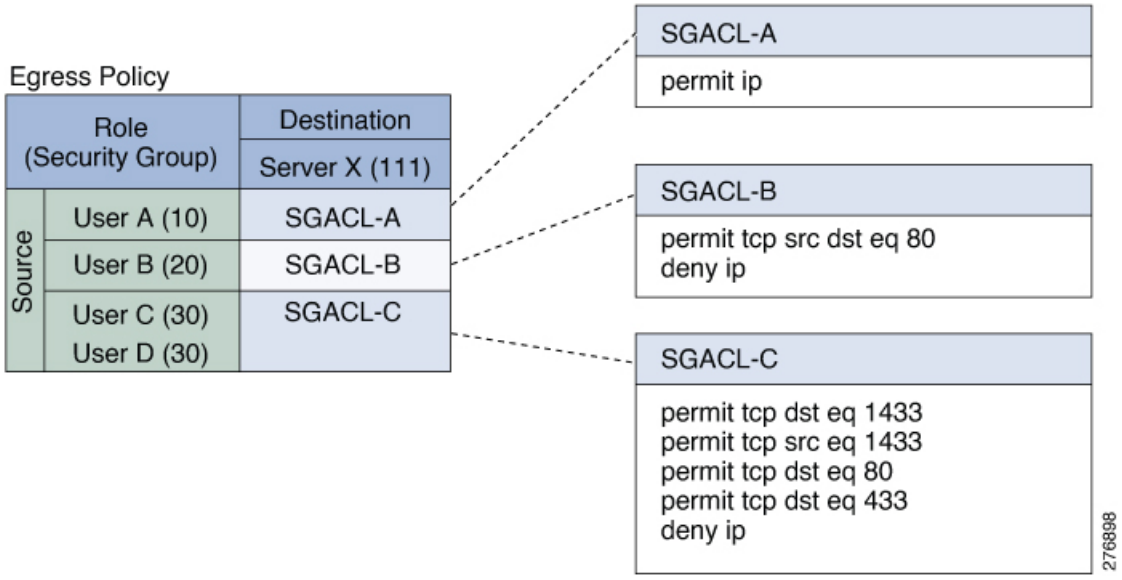
  The device data plane receives the CoA packets from the policy provider (ISE) and applies the policy to the CoA packets. The packets are then forwarded to the device control plane where the next level of policy enforcement happens for the incoming CoA packets. To view the hardware and software policy counter hit information, run the **show cts role-based counters** command in privileged EXEC mode.

SGACL Policies

Using security group access control lists (SGACLs), you can control the operations that users can perform based on the security group assignments of users and destination resources. Policy enforcement within the Cisco TrustSec domain is represented by a permissions matrix, with source security group numbers on one axis and destination security group numbers on the other axis. Each cell in the body of the matrix can contain an ordered list of SGACLs which specifies the permissions that should be applied to packets originating from the source security group and destined for the destination security group.

The following figure shows an example of a Cisco TrustSec permissions matrix for a simple domain with three defined user roles and one defined destination resource. Three SGACL policies control access to the destination server based on the role of the user.

Figure 122: SGACL Policy Matrix Example



By assigning users and devices within the network to security groups and applying access control between the security groups, Cisco TrustSec achieves role-based topology-independent access control within the network. Because SGACLs define access control policies based on device identities instead of IP addresses as in traditional ACLs, network devices are free to move throughout the network and change IP addresses. As long as the roles and the permissions remain the same, changes to the network topology do not change the security policy. When a user is added to the device, you simply assign the user to an appropriate security group and the user immediately receives the permissions of that group.

Note SGACL policies are applied to traffic that is generated between two host devices, not to traffic that is generated from a device to an end host device.

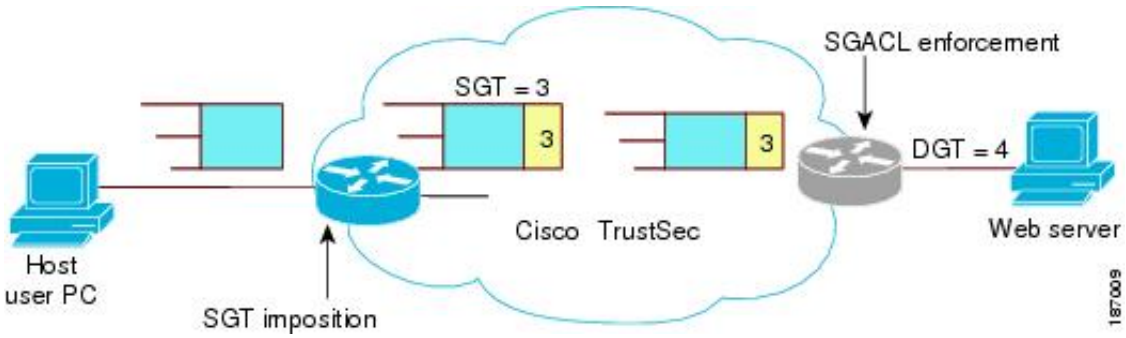
Using role-based permissions greatly reduces the size of ACLs and simplifies their maintenance. With Cisco TrustSec, the number of access control entries (ACEs) configured is determined by the number of permissions specified, resulting in a much smaller number of ACEs than in a traditional IP network. The use of SGACLs in Cisco TrustSec typically results in a more efficient use of TCAM resources compared with traditional ACLs. A maximum of 1,408 SGACL policies are supported.

Ingress Tagging and Egress Enforcement

Cisco TrustSec access control is implemented using ingress tagging and egress enforcement. At the ingress point to the Cisco TrustSec domain, traffic from the source is tagged with an SGT containing the security group number of the source entity. The SGT is propagated with the traffic across the domain. At the egress point of the Cisco TrustSec domain, an egress device uses the source SGT and the security group number of the destination entity (the destination SG, or DGT) to determine which access policy to apply from the SGACL policy matrix.

The following figure shows how the SGT assignment and the SGACL enforcement operate in a Cisco TrustSec domain.

Figure 123: SGT and SGACL in a Cisco TrustSec Domain



1. The host PC transmits a packet to the web server. Although the PC and the web server are not members of the Cisco TrustSec domain, the data path of the packet includes the Cisco TrustSec domain.
2. The Cisco TrustSec ingress device modifies the packet to add an SGT with security group number 3, the security group number assigned by the authentication server for the host PC.
3. The Cisco TrustSec egress device enforces the SGACL policy that applies to source group 3 and destination group 4, the security group number assigned by the authentication server for the web server.
4. If the SGACL allows the packet to be forwarded, the Cisco TrustSec egress switch modifies the packet to remove the SGT and forwards the packet to the web server.

Determining the Source Security Group

A network device at the ingress of Cisco TrustSec domain must determine the SGT of the packet entering the Cisco TrustSec domain so that it can tag the packet with that SGT when it forwards it into the Cisco TrustSec domain. The egress network device must determine the SGT of the packet in order to apply an SGACL.

The network device can determine the SGT for a packet in one of the following methods:

- Obtain the source SGT during policy acquisition—After the Cisco TrustSec authentication phase, a network device acquires policy information from the authentication server, which indicates whether the peer device is trusted or not. If a peer device is not trusted, then the authentication server can also provide an SGT to apply to all packets coming from the peer device.
- Obtain the source SGT from the packet—If a packet comes from a trusted peer device, the packet carries the SGT. This applies to a network device that is not the first network device in Cisco TrustSec domain for the packet.
- Look up the source SGT based on the source identity—With Identity Port Mapping (IPM), you can manually configure the link with the identity of the connected peer. The network device requests policy information, including SGT and trust state, from the authentication server.
- Look up the source SGT based on the source IP address—In some cases, you can manually configure the policy to decide the SGT of a packet based on its source IP address. The SGT Exchange Protocol (SXP) can also populate the IP-address-to-SGT mapping table.

Determining the Destination Security Group

The egress network device in a Cisco TrustSec domain determines the destination group (DGT) for applying the SGACL. The network device determines the destination security group for the packet using the same methods used for determining the source security group, with the exception of obtaining the group number from a packet tag. The destination security group number is not included in a packet tag.

In some cases, ingress devices or other non-egress devices might have destination group information available. In those cases, SGACLs might be applied in these devices rather than egress devices.

SGACL Enforcement on Routed and Switched Traffic

SGACL enforcement is applied only on IP traffic, but enforcement can be applied to either routed or switched traffic.

For routed traffic, SGACL enforcement is performed by an egress switch, typically a distribution switch or an access switch with a routed port connecting to the destination host. When you enable SGACL enforcement globally, enforcement is automatically enabled on every Layer 3 interface except for SVI interfaces.

For switched traffic, SGACL enforcement is performed on traffic flowing within a single switching domain without any routing function. An example would be SGACL enforcement performed by a data center access switch on server-to-server traffic between two directly connected servers. In this example, the server-to-server traffic would typically be switched. SGACL enforcement can be applied to packets switched within a VLAN or forwarded to an SVI associated with a VLAN, but enforcement must be enabled explicitly for each VLAN.

SGACL Logging and ACE Statistics

When logging is enabled in SGACL, the device logs the following information:

- The source security group tag (SGT) and destination SGT
- The SGACL policy name
- The packet protocol type
- The action performed on the packet

The log option applies to individual ACEs and causes packets that match the ACE to be logged. The first packet logged by the log keyword generates a syslog message. Subsequent log messages are generated and reported at five-minute intervals. If the logging-enabled ACE matches another packet (with characteristics identical to the packet that generated the log message), the number of matched packets is incremented (counters) and then reported.

To enable logging, use the **log** keyword in front of the ACE definition in the SGACL configuration. For example, **permit ip log**.

The following is a sample log, displaying source and destination SGTs, ACE matches (for a permit or deny action), and the protocol, that is, TCP, UDP, IGMP, and ICMP information:

```
*Jun 2 08:58:06.489: %C4K_IOSINTF-6-SGACLHIT: list deny_udp_src_port_log-30 Denied
udp 24.0.0.23(100) -> 28.0.0.91(100), SGT8 DGT 12
```

In addition to the existing ‘per cell’ SGACL statistics, which can be displayed using the **show cts role-based counters** command, you can also display ACE statistics, by using the **show ip access-list** *sgacl_name* command. No additional configuration is required for this.

The following example shows how you can use the show ip access-list command to display the ACE count:

```
Device# show ip access-control deny_udp_src_port_log-30

Role-based IP access list deny_udp_src_port_log-30 (downloaded)
10 deny udp src eq 100 log (283 matches)
20 permit ip log (50 matches)
```

Note When the incoming traffic matches the cell, but does not match the SGACL of the cell, the traffic is allowed and the counters are incremented in the HW-Permit for the cell.

The following example shows how the SGACL of a cell works:

The SGACL policy is configured from 5 to 18 with “deny icmp echo” and there is incoming traffic from 5 to 18 with TCP header. If the cell matches from 5 to 18 but traffic does not match with icmp, traffic will be allowed and HW-Permit counter of cell 5 to 18 will get incremented.

```
Device# show cts role-based permissions from 5 to 18

IPv4 Role-based permissions from group 5:sgt_5_Contractors to group
18:sgt_18_data_user2:sgacl_5_18-01
RBACL Monitor All for Dynamic Policies : FALSE
RBACL Monitor All for Configured Policies : FALSE

Device# show ip access-lists sgacl_5_18-01
Role-based IP access list sgacl_5_18-01 (downloaded)
10 deny icmp echo log (1 match)

Device# show cts role-based counters from 5 to 18
Role-based IPv4 counters
From   To   SW-Denied   HW-Denied   SW-Permitt   HW-Permitt   SW-Monitor   HW-Monitor
  5    18       0           0            0        1673202          0            0
```

VRF-aware SGACL Logging

The SGACL system logs will include VRF information. In addition to the fields that are currently logged, the logging information will include the VRF name. The updated logging information will be as shown below:

```
*Nov 15 02:18:52.187: %RBM-6-SGACLHIT_V6: ingress_interface='GigabitEthernet1/1'
sgacl_name='IPV6_TCP_DENY' action='Deny' protocol='tcp' src-vrf='CTS-VRF' src-ip='25::2'
src-port='20'
dest-vrf='CTS-VRF' dest-ip='49::2' dest-port='30' sgt='200' dgt='500'
logging_interval_hits='1'
```

SGACL Monitor Mode

During the pre-deployment phase of Cisco TrustSec, an administrator will use the monitor mode to test the security policies without enforcing them to make sure that the policies function as intended. If the security policies do not function as intended, the monitor mode provides a convenient mechanism for identifying that and provides an opportunity to correct the policy before enabling SGACL enforcement. This enables administrators to have increased visibility to the outcome of the policy actions before they enforce it, and confirm that the subject policy meets the security requirements (access is denied to resources if users are not authorized).

The monitoring capability is provided at the SGT-DGT pair level. When you enable the SGACL monitoring mode feature, the deny action is implemented as an ACL permit on the line cards. This allows the SGACL counters and logging to display how connections are handled by the SGACL policy. Since all the monitored traffic is permitted, there is no disruption of service due to SGACLs while in the SGACL monitor mode.

Authorization and Policy Acquisition

After device authentication ends, both the supplicant and authenticator obtain the security policy from the authentication server. The two peers then perform link authorization and enforce the link security policy against each other based on their Cisco TrustSec device IDs. The link authentication method can be configured as either 802.1X or manual authentication. If the link security is 802.1X, each peer uses a device ID received from the authentication server. If the link security is manual, you must assign the peer device IDs.

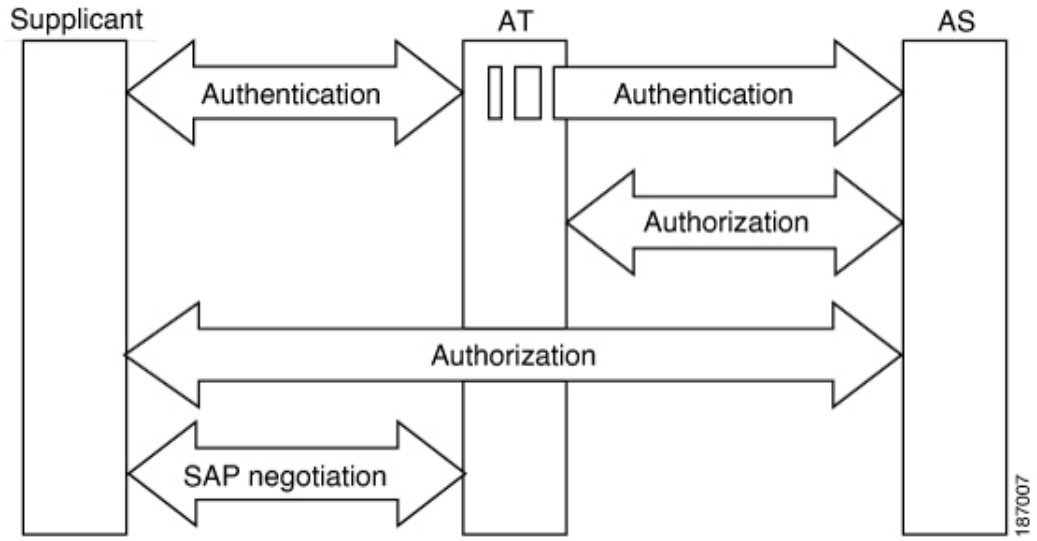
The authentication server returns the following policy attributes:

- Cisco TrustSec trust—Indicates whether the peer device is to be trusted for the purpose of putting the SGT in the packets.
- Peer SGT—Indicates the security group to which the peer belongs. If the peer is not trusted, all packets received from the peer are tagged with this SGT. If the device does not know whether any SGACLs are associated with the peer's SGT, the device may send a follow-up request to the authentication server to download the SGACLs.
- Authorization expiry time—Indicates the number of seconds before the policy expires. A Cisco TrustSec device should refresh its policy and authorization before it times out. The device can cache the authentication and policy data and reuse it after a reboot if the data has not expired.

Note Each Cisco TrustSec device should support some minimal default access policy in case it is not able to contact the authentication server to get an appropriate policy for the peer.

The NDAC and SAP negotiation process is shown in the following figure

Figure 124: NDAC and SAP Negotiation



Environment Data Download

The Cisco TrustSec environment data is a collection of information or policies that assists a device to function as a Cisco TrustSec node. The device acquires the environment data from the authentication server when the device first joins a Cisco TrustSec domain, although you might also manually configure some of the data on a device. For example, you must configure the seed Cisco TrustSec device with the authentication server information, which can later be augmented by the server list that the device acquires from the authentication server.

The device must refresh the Cisco TrustSec environment data before it expires. The device can also cache the environment data and reuse it after a reboot if the data has not expired.

The device uses RADIUS to acquire the following environment data from the authentication server:

- Server lists: List of servers that the client can use for future RADIUS requests (for both authentication and authorization). PAC refresh happens through these servers.
- Device SG: Security group to which the device itself belongs.
- Expiry timeout: Interval that controls how often the Cisco TrustSec device should refresh its environment data.

RADIUS Relay Functionality

The device that plays the role of the Cisco TrustSec authenticator in the 802.1X authentication process has IP connectivity to the authentication server, allowing the device to acquire the policy and authorization from the authentication server by exchanging RADIUS messages over UDP/IP. The supplicant device may not have IP connectivity with the authentication server. In such cases, Cisco TrustSec allows the authenticator to act as a RADIUS relay for the supplicant.

The supplicant sends a special EAPOL message to the authenticator that contains the RADIUS server IP address and UDP port and the complete RADIUS request. The authenticator extracts the RADIUS request from the received EAPOL message and sends it over UDP/IP to the authentication server. When the RADIUS response returns from the authentication server, the authenticator forwards the message back to the supplicant, encapsulated in an EAPOL frame.

Link Security

When both sides of a link support 802.1AE Media Access Control Security (MACsec), a security association protocol (SAP) negotiation is performed. An EAPOL-Key exchange occurs between the supplicant and the authenticator to negotiate a cipher suite, exchange security parameters, and manage keys. Successful completion of all three tasks results in the establishment of a security association (SA).

Depending on your software version, crypto licensing, and link hardware support, SAP negotiation can use one of the following modes of operation:

- Galois/Counter Mode (GCM)—Specifies authentication and encryption
- GCM authentication (GMAC)—Specifies authentication and no encryption
- No Encapsulation—Specifies no encapsulation (clear text)
- Null—Specifies encapsulation, no authentication and no encryption

All modes except No Encapsulation require Cisco TrustSec-capable hardware.

Configuring SAP-PMK for Link Security

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Configures an interface and enters interface configuration mode.
Step 4	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Specifies a trunking VLAN Layer 2 interface.

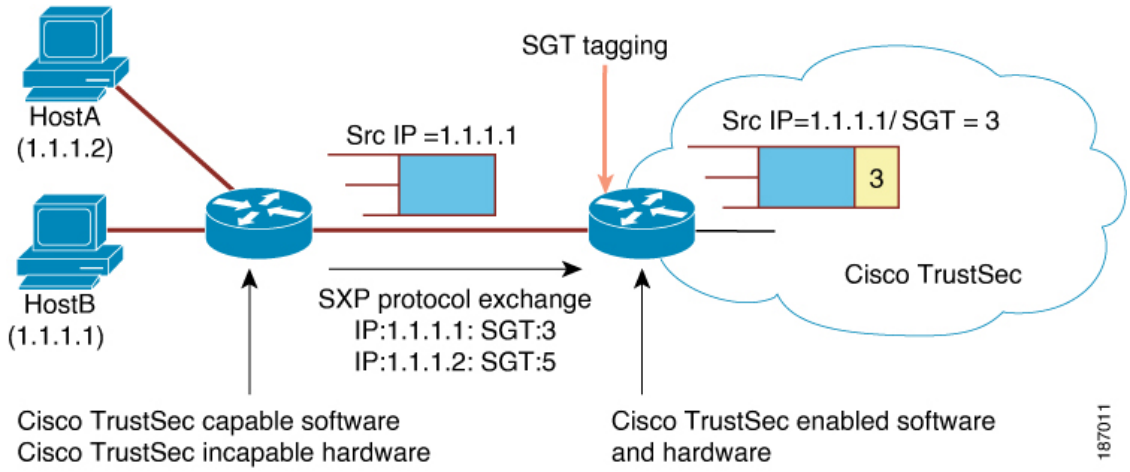
	Command or Action	Purpose
Step 5	**cts manual** **Example:** `Device(config-if)# cts manual`	Enters Cisco TrustSec manual configuration mode.
Step 6	**no propagate sgt** **Example:** `Device(config-if-cts-manual)# no propagate sgt`	Use the **no** form of this command when the peer is incapable of processing a SGT. The **no propagate sgt** command prevents the interface from transmitting the SGT to the peer.
Step 7	**sap pmk** *key* [**mode-list** *mode1* [*mode2* [*mode3* [*mode4*]]]] **Example:** `Device(config-if-cts-manual)# sap pmk 00000000000000000000000000000000000000000000000000000000000000001234567890 mode-list gcm-encrypt gmac`	Configures the SAP pairwise master key (PMK) and operation mode. SAP is disabled by default in Cisco TrustSec manual mode. • *key*: A hexadecimal value with an even number of characters and a maximum length of 32 characters. The SAP operation mode options are described below: • **gcm-encrypt**: Authentication and encryption **Note** Select this mode for MACsec authentication and encryption if your software license supports MACsec encryption. • **gmac**: Authentication, no encryption • **no-encap**: No encapsulation • **null**: Encapsulation, no authentication or encryption **Note** If the interface is not capable of data link encryption, the **no-encap** command is the default and the only available SAP operating mode. SGT is not supported.
Step 8	**end** **Example:** `Device(config-if-cts-manual)# end`	Exits Cisco TrustSec manual configuration mode and returns to privileged EXEC mode..

SXP for SGT Propagation Across Legacy Access Networks

Tagging packets with SGTs requires hardware support. You might have devices in your network that, while capable of participating in Cisco TrustSec authentication, lack the hardware capability to tag packets with SGTs. By using the SGT Exchange Protocol (SXP), these devices can pass IP-address-to-SGT mappings to a Cisco TrustSec peer device that has Cisco TrustSec-capable hardware.

SXP typically operates between ingress access layer devices at the Cisco TrustSec domain edge and distribution layer devices within the Cisco TrustSec domain. The access layer device performs Cisco TrustSec authentication of external source devices to determine the appropriate SGTs for ingress packets. The access layer device learns the IP addresses of the source devices using IP device tracking and (optionally) DHCP snooping, then uses SXP to pass the IP addresses of the source devices along with their SGTs to the distribution devices. Distribution devices with Cisco TrustSec-capable hardware can use this IP-to-SGT mapping information to tag packets appropriately and to enforce SGACL policies.

Figure 125: SXP Protocol to Propagate SGT Information



You must manually configure an SXP connection between a peer without Cisco TrustSec hardware support and a peer with Cisco TrustSec hardware support. The following tasks are required when configuring the SXP connection:

- If you require SXP data integrity and authentication, you must configure the same SXP password on both peer devices. You can configure the SXP password either explicitly for each peer connection or globally for the device. Although an SXP password is not required, we recommend its use.
- You must configure each peer on the SXP connection as either an SXP speaker or an SXP listener. The speaker device distributes the IP-to-SGT mapping information to the listener device.
- You can specify a source IP address to use for each peer relationship or you can configure a default source IP address for peer connections where you have not configured a specific source IP address. If you do not specify any source IP address, the device will use the interface IP address of the connection to the peer.

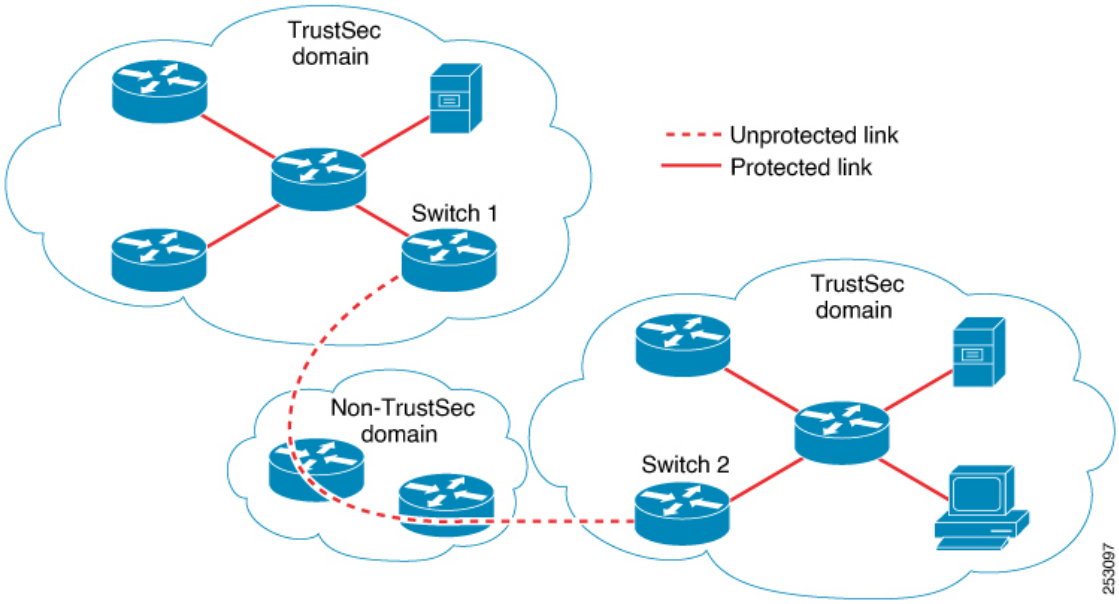
SXP allows multiple hops. That is, if the peer of a device lacking Cisco TrustSec hardware support also lacks Cisco TrustSec hardware support, the second peer can have an SXP connection to a third peer, continuing the propagation of the IP-to-SGT mapping information until a hardware-capable peer is reached. A device can be configured as an SXP listener for one SXP connection as an SXP speaker for another SXP connection.

A Cisco TrustSec device maintains connectivity with its SXP peers by using the TCP keepalive mechanism. To establish or restore a peer connection, the device will repeatedly attempt the connection setup using a configurable retry period until the connection is successful or until the connection is removed from the configuration.

Layer 3 SGT Transport for Spanning Non-TrustSec Regions

When a packet leaves the Cisco TrustSec domain for a non-TrustSec destination, the egress Cisco TrustSec device removes the Cisco TrustSec header and SGT before forwarding the packet to the outside network. If, however, the packet is merely traversing a non-TrustSec domain on the path to another Cisco TrustSec domain, as shown in the following figure, the SGT can be preserved by using the Cisco TrustSec Layer 3 SGT Transport feature. In this feature, the egress Cisco TrustSec device encapsulates the packet with an ESP header that includes a copy of the SGT. When the encapsulated packet arrives at the next Cisco TrustSec domain, the ingress Cisco TrustSec device removes the ESP encapsulation and propagates the packet with its SGT.

Figure 126: Spanning a Non-TrustSec domain



To support Cisco TrustSec Layer 3 SGT Transport, any device that will act as a Cisco TrustSec ingress or egress Layer 3 gateway must maintain a traffic policy database that lists eligible subnets in remote Cisco TrustSec domains as well as any excluded subnets within those regions. You can configure this database manually on each device if they cannot be downloaded automatically from the Cisco Secure ACS.

A device can send Layer 3 SGT Transport data from one port and receive Layer 3 SGT Transport data on another port, but both the ingress and egress ports must have Cisco TrustSec-capable hardware.

Note Cisco TrustSec does not encrypt the Layer 3 SGT Transport encapsulated packets. To protect the packets traversing the non-TrustSec domain, you can configure other protection methods, such as IPsec.

VRF-Aware SXP

The SXP implementation of Virtual Routing and Forwarding (VRF) binds an SXP connection with a specific VRF. It is assumed that the network topology is correctly configured for Layer 2 or Layer 3 VPNs, with all VRFs configured before enabling Cisco TrustSec.

SXP VRF support can be summarized as follows:

- Only one SXP connection can be bound to one VRF.
- Different VRFs may have overlapping SXP peer or source IP addresses.
- IP–SGT mappings learned (added or deleted) in one VRF can be updated only in the same VRF domain. The SXP connection cannot update a mapping bound to a different VRF. If no SXP connection exits for a VRF, IP–SGT mappings for that VRF won't be updated by SXP.
- Multiple address families per VRF is supported. Therefore, one SXP connection in a VRF domain can forward both IPV4 and IPV6 IP-SGT mappings.
- SXP has no limitation on the number of connections and number of IP–SGT mappings per VRF.

Layer 2 VRF-Aware SXP and VRF Assignment

VRF to Layer 2 VLANs assignments are specified with the **cts role-based l2-vrf vrf-name vlan-list** global configuration command. A VLAN is considered a Layer 2 VLAN as long as there is no switch virtual interface (SVI) with an IP address configured on the VLAN. The VLAN becomes a Layer 3 VLAN once an IP address is configured on its SVI.

The VRF assignments configured by the **cts role-based l2-vrf** command are active as long as a VLAN remains a Layer 2 VLAN. The IP–SGT bindings learned while a VRF assignment is active are also added to the Forwarding Information Base (FIB) table associated with the VRF and the IP protocol version. If an SVI becomes active for a VLAN, the VRF to VLAN assignment becomes inactive and all the bindings learned on the VLAN are moved to the FIB table associated with the SVI's VRF.

The VRF to VLAN assignment is retained even when the assignment becomes inactive. It is reactivated when the SVI is removed or when the SVI IP address is deconfigured. When reactivated, the IP–SGT bindings are moved back from the FIB table associated with the SVI's VRF to the FIB table associated with the VRF assigned by the **cts role-based l2-vrf** command.

CHAPTER 118

SGACL and Environment Data Download over REST

This module describes the downloading of SGACL and environment data over REST APIs.

Prerequisites for SGACL and Environment Data Download over REST

- Cisco Identity Services Engine (ISE) Version should be 2.7 and above.
- The network device configuration on Cisco ISE must be updated to include the configuration to allow REST API calls from a network device IP address (NAS-IP). The device ID and password specified in the Cisco ISE configuration is included as the username and password by the network device that makes REST API calls to Cisco ISE.

Restrictions for SGACL and Environment Data Download over REST

- Cisco TrustSec Change of Authorization (CoA) uses RADIUS as the protocol.
- Only port 9063 is supported as the ERS server port.
- Server statistics is not persistent after a refresh of the environment data.
- Only one Fully Qualified Domain Name (FQDN) per server is supported.

- For RADIUS, policy download over IPv6 server is not supported.

Information About SGACL and Environment Data Download over REST

SGACL and Environment Data Download over REST Overview

Cisco TrustSec uses the REST-based transport protocol for policy provisioning and environment data download from Cisco Identity Services Engine (ISE). The REST-based protocol is more secure, and provides reliable, and faster Security Group access control list (SGACL) policy and environment data provisioning, than older RADIUS protocols.

Both the REST API-based and RADIUS-based download of Cisco TrustSec data is supported. However, only one protocol can be active on a device. REST-based protocol is the default, however, you can change the protocol to RADIUS by configuring the **cts authorization list** command.

Note Cisco TrustSec Change of Authorization (CoA) will still use RADIUS as the protocol.

Cisco TrustSec Security Group Access Control List (SGACL) and environment data are synchronized from the active device to the standby device, after the policy is installed. However, REST API connections or sessions are not synchronized during a switchover.

8 IPv4 and 8 IPv6 addresses are supported per server. Cisco TrustSec device honors the 429 response code from Cisco ISE. This response code is sent by Cisco ISE, when it is overloaded. Once a 429 response code is received for a particular server, the device marks the server as dead, and switches to the next server in the list (private or public). The next retry attempt is done after 60 seconds.

Cisco TrustSec Environment Data

Environment data comprises of operational data that supplement Cisco TrustSec functions. The environment data request from a device to Cisco ISE consists of the following data:

- Device name: Specifies the name of the device.
- Device capability: Specifies additional data.

The environment data response from Cisco ISE to a device consists of the following data:

- Device security group tag (SGT): Derived from Cisco ISE based on the device name.
- Server list: Displays the list of Cisco TrustSec servers specified in Cisco ISE.
- SG-Name Table: Displays the mapping between SGT and the device name. SGT is displayed in numerals and the device name in text format.
- Refresh time: Indicates the time when the environment data will be refreshed.

Note

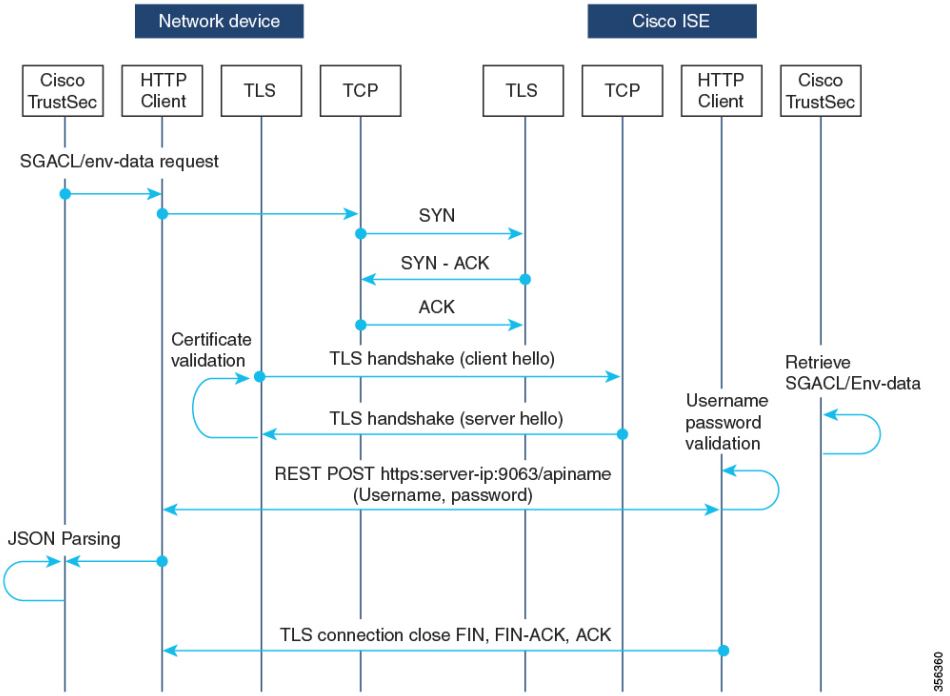
- As part of Cisco TrustSec environment data refresh, the last received servers are deleted and newly received servers are added to the server list. As a result of the refresh, the server list statistics restarts from zero, and the server status is set to *Inactive* and the IP address state is set to *Reachable*. The device then updates the server statistics and status based on the subsequent policy request and response.
- You can configure automated tester to be VRF aware. You can use the **vrf** keyword with the **automate-tester** command to enable automate-tester for a non-default VRF.

  For VRF aware automate-tester to work, you have to configure **global config ipv4/ipv6 source interface** *interface-name* **vrf** *vrf-name* command.

Message Flow Between a Network Device and a Server

The following illustration displays the connection management for REST calls between a network device and server.

Figure 127: Message Flow Between a Network Device and a Server



- Cisco ISE REST API service runs on a secure socket that runs Transport Layer Security (TLS) 1.2 server on port 9063 to service network device requests for SGACL and environment data.
- The device uses a make or break approach to the TLS connection establishment, and there is no persistent TLS connection between the device and Cisco ISE. After the TLS connection is established, the device can use this connection to submit multiple REST API calls to specific resource uniform resource locators (URLs). After all REST requests are processed, the server terminates the connection through a TCP-FIN message. For new REST API calls a new connection must be established with the server.

- The REST API call from the device to Cisco ISE starts with a TCP connection establishment. Cisco ISE must be configured with device IP address to allow ingress connections from the device. TCP connection requests from source IP addresses that are not configured on Cisco ISE are dropped, and an audit log created.
- Username and password: Every RESTAPI call must include the username and password authentication while requesting access to a resource uniform resource identifier (URI). The authentication helps the server to determine if the caller should be given access to the resource or to deny the request.
- A successful TLS connection establishment with Cisco ISE requires its server-certificate signature or PEM to be installed as the trustpoint (by using the **crypto pki trustpoint** command) on the device to trust the server. Only fingerprint or signature of the server certificate need to be exported and installed on the device under a trustpoint. Import of private-key of the server certificate is not necessary.
- After establishing the TLS connection, the HTTP client on the device initiates a REST call to Cisco ISE on the specified resource.

Policy Server Selection Criteria

Multiple HTTP policy servers are configured on a Cisco TrustSec device. Once a server is selected, the device use this server to interact with Cisco ISE until the server is marked as dead.

There are two types of server selection:

- In-Order Selection: This is the default behavior, where servers are picked in the order in which they are configured (from the public server list) or downloaded (from the private server list). Once a server is selected, the device is used till it is marked as dead, and then the next server in the list is selected.

  When environment data is successfully downloaded, and a server-list is available, these servers are added to the private server list.

- Random Server-Selection: When multiple HTTP policy servers are configured on a device, a single Cisco ISE instance may get overloaded if the device always selects the first configured server. To avoid this situation, each device will randomly select a server. A random number is generated by the device and based on this number a server is selected. For different devices to generate random numbers, the unique board ID and the Cisco TrustSec process ID of the device is used to initialize the random number generator.

  Once a server is selected, all future requests go this server until the server is marked as dead. Once a server is in the dead state, the random server selection logic picks up the next alive server. The dead server is not added to the count of active servers when picking the new server. The server numbering starts with zero.

  Selected Server = (Generated Random Number) % (Total Number of Active Servers).

To change the server selection logic to random, use the **cts policy-server order random** command.

Server and IP Address Selection Process

The order of server-selection is the private server-list (received as part of server-list download), followed by the public server-list (configured servers). Within these server lists, the order can either be random selection or in-order selection based on whether the **cts policy-server order random** command is enabled or not.

Multiple IP (both IPv4 and IPv6) addresses per server are supported. The order of IP selection is IPv4 addresses, followed by IPv6 addresses, and then FQDN.

This section describes how the server and IP address selection works:

1. When a device boots up for the first time, a server from the public (configured) list is selected.
2. If the **cts environment-data enable** command is configured, the device uses the public server to download the private server-list from Cisco ISE.
3. After successfully receiving the private list, all subsequent requests will use the private list.
4. After the server and IP address are selected, the device connects to Cisco ISE using the server/IP address combination. This server will interact with Cisco ISE until it fails to get a response.
5. If no response is received from the current active server in the private list, the device switches to the next server in the list. If the server is selected for the first time, the IP selection logic searches for the first reachable IP or IPv6 address.
6. After the server and IP address selection, the device is used until it goes down.
7. If none of the servers in the private list are reachable, the device attempts to connect to the servers in the public list. The server switching logic and IP selection are the same for private and public list.
8. The server change happens only when the server list is refreshed.
9. If all servers in both the private and public server list are dead, the device restarts the server/IP address selection logic from the start of the private list.
10. When a specific server/IP address combination fails, the device waits for 60 seconds before it attempts a new combination.

Server Liveliness Check

Whether a server is alive is determined after sending an environment-data or an SGACL request to Cisco ISE. There is no liveliness detection phase after a server is configured or downloaded as part of a server list. The default server status is alive for all types of servers.

When a request is sent to Cisco ISE, and if the server is not reachable or the response is lost, the server is moved to dead state. The server selection logic will pick the same server and the next IP address (if multiple addresses are configured) to send the next set of Cisco ISE requests. The logic will pick the next server in the list, if the device receives the overloaded response (HTTP 429) from Cisco ISE.

A server can be marked as dead because of any of the following reasons:

- The configured IP address is not reachable.
- Incorrect port number.
- The Cisco ISE instance with the IP address is down.
- The interface towards Cisco ISE is down.
- A Transport Layer Security (TLS) handshake failure.
- An HTTP response timeout.
- An incorrectly configured domain name (if a domain name is used).

If a server has both the static IP address and the domain name configured, preference is given to the static IP address. If there is no response to the static IP address, the device tries with the domain name. When no response is received with both the static IP address and the domain-name, the server is marked as dead.

When all servers of the private list are marked as dead, the device uses the public list. If all remaining servers are also marked as dead, then the recovery mechanism starts. The device waits for the next Cisco TrustSec request (for policy refresh, environment data download or refresh, and so on), and marks all the servers as alive to retry the download. If there is no trigger for a new Cisco TrustSec request, the servers remain in the dead state.

How to Configure SGACL and Environment Data Download over REST

Configuring the Username and Password

Configure the username and password in Cisco ISE as the REST API access credentials, before configuring it on the device. See the Cisco TrustSec HTTP Servers section of the "Cisco TrustSec Policies Configuration" chapter for more information.

Note If you try to configure RADIUS-based configuration by using the **cts authorization-list** command, when the HTTP-based configurations are already enabled, the following error message is displayed on the console:

```
Error: 'cts policy-server or cts environment-data' related configs are enabled.
Disable http-based configs, to enable 'cts authorization'
```

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts policy-server name** *server-name* **Example:** `Device(config)# cts policy-server name ISE-server`	Configures a Cisco TrustSec policy server and enters policy-server configuration mode.
Step 4	**exit** **Example:** `Device(config-policy-server)# exit`	Exits policy-server configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 5	**cts policy-server username** *username* **password** {**0** | **6** |**7** | *password*} {*password*} **Example:** `Device(config)# cts policy-server username admin password 6 password1`	Configures an username and password. **Note** This username and password must be created on Cisco ISE as the REST API access credentials before configuring it on the device.
Step 6	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Certificate Enrollment

Third-party Certificate Authority (CA) certificate and chain of certificates are supported. Perform the following steps to enrol a certificate:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Device(config)# crypto pki trustpoint mytp`	Declares the trustpoint and a given name, and enters ca-trustpoint configuration mode.
Step 4	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 5	**crypto pki authenticate** *name* **Example:** `Device(config)# crypto pki authenticate mytp`	Retrieves the Certificate Authority (CA) certificate and authenticates it. Check the certificate fingerprint if prompted. **Note** This command is optional if the CA certificate is already loaded into the configuration.

	Command or Action	Purpose
Step 6	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Downloading Cisco TrustSec Policies

The **cts role-based enforcement** must already be configured to download Cisco TrustSec Policies.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts policy-server name** *server-name* **Example:** `Device(config)# cts policy-server name ISE-server`	Configures a Cisco TrustSec policy server and enters policy-server configuration mode.
Step 4	**address domain-name** *name* **Example:** `Device(config-policy-server)# address domain-name domain1`	Configures the domain name address of the policy server.
Step 5	**address** {**ipv4**| **ipv6** *policy-server-address* **Example:** `Device(config-policy-server)# address ipv4 10.1.1.1` `Device(config-policy-server)# address ipv6 2001.DB8::1`	Configures the IPv4 or IPv6 address of the policy server.
Step 6	**tls server-trustpoint** *name* **Example:** `Device(config-policy-server)# tls server-trustpoint tls1`	Configures the Transport Layer Security trustpoint.
Step 7	**timeout** *seconds* **Example:** `Device(config-policy-server)# timeout 15`	(Optional) Configures the response timeout in seconds. • The default is 5 seconds.

	Command or Action	Purpose
Step 8	**retransmit** *number-of-retries* **Example:** `Device(config-policy-server)# retransmit 4`	(Optional) Configures the maximum number of retries from the server. • The default is 4.
Step 9	**port** *port-number* **Example:** `Device(config-policy-server)# port 9063`	(Optional) Configures the policy server port number. **Note** The ERS server port number must be 9063. You cannot change this port number.
Step 10	**content-type json** **Example:** `Device(config-policy-server)# content-type json`	(Optional) Configures the content type to source SGACL and environment data from Cisco ISE. **Note** By default, JSON is used as the content type, even if this command is not configured.
Step 11	**end** **Example:** `Device(config-policy-server)# end`	Exits policy-server configuration mode and returns to privileged EXEC mode.

Downloading Environment Data

The source interface to use for HTTP connections must be specified in the **ip http client source-interface** command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts policy-server device-id** *device-ID* **Example:** `Device(config)# cts policy-server device-id server1`	Configures the policy server device ID to send environment data requests to Cisco ISE. • This device-ID must be the one used to add the network access device (NAD) on Cisco ISE.

	Command or Action	Purpose
Step 4	**cts environment-data enable** **Example:** `Device(config)# cts environment-data enable`	Enables the downloading of environment data from Cisco ISE. **Note** The **cts environment-data enable** command and the **cts authorization list** command are mutually exclusive. These commands cannot be configured together.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Verifying the SGACL and Environment Data Download over REST

Use the following commands in any order:

- **show cts policy-server details name**

  Displays information about the specified policy server.

```
Device# show cts policy-server details name ise_server_1

Server Name   : ise_server_1
Server Status : Active
  IPv4 Address     : 10.64.69.84
  IPv6 Address     : 2001:DB::2
  Trustpoint       : ISE84
  Port-num         : 9063
  Retransmit count : 3
  Timeout          : 15
  App Content type : JSON
```

- **show cts policy-server statistics active**

  Displays statistics information about active policy servers.

  When you use the command without the **active** the statistics of all servers are listed.

```
Device# show cts policy-server statistics active

Server Name  : ise_server_1
Server State : ALIVE
  Number of Request sent        : 7
  Number of Request sent fail   : 0
  Number of Response received   : 4
  Number of Response recv fail  : 3
    HTTP 200 OK                 : 4
    HTTP 400 BadReq             : 0
    HTTP 401 UnAuthorized Req   : 0
    HTTP 403 Req Forbidden      : 0
    HTTP 404 NotFound           : 0
    HTTP 408 ReqTimeout         : 0
    HTTP 415 UnSupported Media  : 0
    HTTP 500 ServerErr          : 0
```

```
    HTTP 501 Req NoSupport      : 0
    HTTP 503 Service Unavailable: 0
    TCP or TLS handshake error  : 3
    HTTP Other Error            : 0
```

- **show cts server-list**

  Displays the list of servers that are downloaded as part of the environment data. These servers will be part of private server-list.

Note The following output displays the HTTP-based download information:

```
Device# show cts server-list

HTTP Server-list:
  Server Name      : cts_private_server_0
  Server State     : ALIVE
  IPv4 Address     : 10.64.69.151
  IPv6 Address     : 2001:DB8:8086:6502::
  IPv6 Address     : 2001:db8::2
  IPv6 Address     : 2001:db8::402:99
  IPv6 Address     : 2001:DB8:4::802:16
  Domain-name      : ise-267.cisco.com
  Trustpoint       : cts_trustpoint_0

  Server Name      : cts_private_server_1
  Server State     : ALIVE
  IPv4 Address     : 10.10.10.3
  IPv4 Address     : 10.10.10.2
  IPv6 Address     : 2001:DB8::20
  IPv6 Address     : 2001:DB8::21
  Domain-name      : www.ise.cisco.com
  Trustpoint       : cts_trustpoint_1
```

Debugging the SGACL and Environment Data over REST Configuration

Use the following **debug** commands for debugging the configuration.

- **debug cts policy-server http**

  Enables HTTP client debugging.

- **debug cts policy-server json**

  Enables JSON client debugging.

Configuration Examples for SGACL and Environment Data Download over REST

Example: Configuring the Username and Password

```
Device> enable
Device# configure terminal
Device(config)# cts policy-server name ISE-server
Device(config-policy-server)# exit
Device(config)# cts policy-server username admin 6 password1
Device(config)# end
```

Example: Downloading Cisco TrustSec Policies

```
Device> enable
Device# configure terminal
Device(config)# cts role-based enforcement
Device(config)# cts policy-server name ISE-server
Device(config-policy-server)# address domain-name domain1
Device(config-policy-server)# address ipv4 10.1.1.1
Device(config-policy-server)# address ipv6 2001:DB8::1
Device(config-policy-server)# tls server-trustpoint tls1
Device(config-policy-server)# timeout 15
Device(config-policy-server)# retransmit 4
Device(config-policy-server)# port 2010
Device(config-policy-server)# end
```

Example: Downloading Environment Data

```
Device> enable
Device# configure terminal
Device(config)# cts policy-server name ISE-server
Device(config-policy-server)# exit
Device(config)# cts policy-server device-id server1
Device(config)# cts env-data enable
Device(config)# end
```

CHAPTER 119

Configuring Security Group ACL Policies

Using security group access control lists (SGACLs), you can control the operations that users can perform based on the security group assignments of users and destination resources. Policy enforcement within the Cisco TrustSec domain is represented by a permissions matrix, with source security group numbers on one axis and destination security group numbers on the other axis. Each cell in the body of the matrix can contain an ordered list of SGACLs, which specifies the permissions that should be applied to packets originating from the source security group and destined for the destination security group.

Restrictions for Configuring Security Group ACL Policies

- Due to hardware limitations, Cisco TrustSec SGACLs cannot be enforced for punt (CPU bound) traffic in hardware. SGACL enforcement in software is bypassed wfor CPU-bound traffic for switch virtual interface (SVI) and Layer 2 and Layer 3 Location Identifier Separation Protocol (LISP), and loopback interfaces.
- When configuring SGACL policies, if you change the IP version dynamically from **IPv4** or **IPv6** to **Agnostic** (applies to both IPv4 and IPv6) and vice-versa, the corresponding SGACL policies for IPv4 and IPv6 are not downloaded completely through the management VRF interface.
- When configuring SGACL policies, if you change the existing IP version to any other version (**IPv4**, **IPv6**, or **Agnostic**) and vice-versa, Change of Authorization (CoA) from Cisco Identity Services Engine (ISE) cannot be performed using RADIUS. Instead, use SSH and run the **cts refresh policy** command to perform a manual policy refresh.
- When using an allowed SGT model with default action as **deny all**, in some cases, Cisco TrustSec policies are only partially downloaded from the ISE server after a device reload.

  To prevent this, define a static policy on the device. Even if the **deny all** option is applied, the static policy permits traffic that allows the device to download policies from the ISE server and overwrite the defined static policies. For device SGT, configure the following commands in global configuration mode:

  - **cts role-based permissions from** <*sgt_num*> **to unknown**
  - **cts role-based permissions from unknown to** <*sgt_num*>

Information About Security Group ACL Policies

The following sections provide information about configuring SGACL policies.

SGACL Logging

A device can provide logging messages about packets that are permitted or denied by a standard IP access list. That is, any packet that matches an SGACL causes an informational logging message about the packet to be sent to the console. The limit of messages logged to the console is controlled by the **logging console** command that controls the syslog messages. The SGACL logging has been enhanced to use NetFlow hardware, which allows much larger logging rates.

Note SGACL logging in hardware is only supported for Role-Based access control list (RBACL).

The first packet that triggers the SGACL creates a flow, and logging is done at the NetFlow timeout of 30 seconds and 1 minute for inactive and active flows respectively. Subsequent packets are collected over 5-minute intervals before they are logged. The logging message includes the access list number, whether the packet was permitted or denied, the source and destination IP addresses of the packet, the interface on which the packet was ingressed, and the number of packets from that source permitted or denied in the previous 5-minute interval.

Note

- Because SGACL logging in the hardware is done using NetFlow, if a NetFlow-based feature is applied to an interface, logging for that interface falls back to the old mechanism. Logging through NetFlow hardware starts again for that interface after the NetFlow-based feature is removed. The rest of the interfaces continue logging through NetFlow hardware.
- Only 15 NetFlow monitors can be attached to the device at a given time. SGACL logging requires one NetFlow monitor each for IPv4 and IPv6 logging. If NetFlow monitors are not available for logging, SGACL logging is done through the earlier mechanism. Once the required number of NetFlow monitors are available, run the **cts role-based permissions** command to trigger logging through the NetFlow hardware again.
- If a log access control entry (ACE) has fields other than source port number, destination port number and the protocol in use, logging is done through the earlier mechanism.

How to Configure Security Group ACL Policies

The following sections provide information about various SGACL policy configurations.

SGACL Policy Configuration Process

Follow these steps to configure and enable SGACL policies:

1. Configuration of SGACL policies should be done primarily through the Policy Management function of the Cisco Secure Access Control Server (ACS) or the Cisco Identity Services Engine (ISE).

   If you are not using AAA on a Cisco Secure ACS or a Cisco ISE to download the SGACL policy configuration, you can manually configure the SGACL mapping and policies.

Note An SGACL policy that is downloaded dynamically from the Cisco Secure ACS or a Cisco ISE will override any conflicting locally-defined policy.

2. To enable SGACL policy enforcement on egress traffic on routed ports, enable SGACL policy enforcement globally as described in the *Enabling SGACL Policy Enforcement Globally* section.
3. To enable SGACL policy enforcement on switched traffic within a VLAN, or on traffic that is forwarded to an SVI that is associated with a VLAN, enable SGACL policy enforcement for specific VLANs, as described in the *Enabling SGACL Policy Enforcement on VLANs* section.

Enabling SGACL Policy Enforcement Globally

You must enable SGACL policy enforcement globally for Cisco TrustSec-enabled routed interfaces.

To enable SGACL policy enforcement on routed interfaces, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts role-based enforcement** **Example:** `Device(config)# cts role-based enforcement`	Enables Cisco TrustSec SGACL policy enforcement on routed interfaces.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Enabling SGACL Policy Enforcement Per Interface

You must first enable SGACL policy enforcement globally for Cisco TrustSec-enabled routed interfaces. This feature is not supported on port channel interfaces.

To enable SGACL policy enforcement on Layer 3 interfaces, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot/port* **Example:** `Device(config)# interface gigabitethernet 1/1`	Configures an interface and enters interface configuration mode.
Step 4	**cts role-based enforcement** **Example:** `Device(config-if)# cts role-based enforcement`	Enables Cisco TrustSec SGACL policy enforcement on routed interfaces.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 6	**show cts interface** **Example:** `Device# show cts interface`	(Optional) Displays Cisco TrustSec states and statistics per interface.

Enabling SGACL Policy Enforcement on VLANs

You must enable SGACL policy enforcement on specific VLANs to apply access control to switched traffic within a VLAN, or to traffic that is forwarded to an SVI associated with a VLAN.

To enable SGACL policy enforcement on a VLAN or a VLAN list, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts role-based enforcement vlan-list** *vlan-list* **Example:** `Device(config)# cts role-based enforcement vlan-list 31-35,41`	Enables Cisco TrustSec SGACL policy enforcement on the VLAN or VLAN list.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring SGACL Monitor Mode

Before configuring SGACL monitor mode, ensure the following:

- Cisco TrustSec is enabled
- Counters are enabled

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts role-based monitor all** **Example:** `Device(config)# cts role-based monitor all`	Enables global monitor mode.

	Command or Action	Purpose
Step 4	**cts role-based monitor permissions from** {*sgt_num*} **to** {*dgt_num*} [**ipv4** | **ipv6**] **Example:** `Device(config)# cts role-based permissions from 2 to 3 ipv4`	Enables monitor mode for IPv4 or IPv6 Role-Based Access Control List (RBACL) (Security Group Tag-Destination Group Tag [SGT-DGT] pair).
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	**show cts role-based permissions from** {*sgt_num*} **to** {*dgt_num*} [**ipv4** | **ipv6**] [**details**] **Example:** `Device# show cts role-based permissions from 2 to 3 ipv4 details`	(Optional) Displays the SGACL policies and details about the monitor mode functionality for each pair. The command output displays if per-cell monitor mode is enabled for the <*SGT-DGT*> pair.
Step 7	**show cts role-based counters** [**ipv4** | **ipv6**] **Example:** `Device# show cts role-based counters ipv4`	(Optional) Displays all the SGACL enforcement statistics for IPv4 and IPv6 events.

Manually Configuring SGACL Policies

A role-based access control list bound to a range of SGTs and DGTs forms an SGACL, a Cisco TrustSec policy enforced on egress traffic. Configuration of SGACL policies are best done through the policy-management functions of Cisco ISE or Cisco Secure ACS. To manually, that is, locally, configure SGACL policies, configure a role-based ACL and bind this role-based ACL to a range of SGTs.

Note An SGACL policy downloaded dynamically from Cisco ISE or Cisco ACS overrides conflicting manually configured policies, if any.

Configuring and Applying IPv4 SGACL Policies

Note When configuring SGACLs and RBACLs, the named access control lists (ACLs) must start with an alphabet.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip access-list role-based** *rbacl-name* **Example:** `Device(config)# ip access-list role-based allow_webtraff`	Creates an RBACL and enters Role-based ACL configuration mode.
Step 4	{[*sequence-number*] | **default** | **permit** | **deny** | **remark**} **Example:** `Device(config-rb-acl)# 10 permit tcp dst eq 80 dst eq 20`	Specifies the access control entries (ACEs) for the RBACL. You can use most of the commands and options allowed in extended named access list configuration mode, with the source and destination fields omitted. The following ACE keywords are not supported: • **reflect** • **evaluate** • **time-range**
Step 5	**exit** **Example:** `Device(config-rb-acl)# exit`	Exits role-based ACL configuration mode and returns to global configuration mode.
Step 6	**cts role-based permissions {default** | [**from** {*sgt_num* | **unknown**} **to** {*dgt_num* | **unknown** }] {*rbacls* | **ipv4** *rbacls*} **Example:** `Device(config)# cts role-based permissions from 55 to 66 allow_webtraff`	Binds SGTs and DGTs to the RBACL. The configuration is analogous to populating the permission matrix configured on Cisco ISE or Cisco Secure ACS. • **default**: Default permissions list. • *sgt_num*: 0 to 65,519. Source Group Tag. • *dgt_num*: 0 to 65,519. Destination Group Tag. • **unknown**: SGACL applies to packets where the security group (source or destination) cannot be determined. • **ipv4**: Indicates the RBACLs are IPv4. • *rbacls*: Names of RBACLs.
Step 7	**end** **Example:**	Exits global configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
Step 8	**show cts role-based permissions** **Example:** `Device# show cts role-based permissions`	(Optional) Displays permission to RBACL configurations.
Step 9	**show ip access-lists** {*rbacls* | **ipv4** *rbacls*} **Example:** `Device# show ip access-lists allow_webtraff`	(Optional) Displays ACEs of all RBACLs or a specified RBACL.

Configuring IPv6 SGACL Policies

To manually configure IPv6 SGACL policies, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list role-based** *sgacl-name* **Example:** `Device(config)# ipv6 access-list role-based sgaclname`	Creates a named IPv6 SGACL and enters IPv6 role-based ACL configuration mode.
Step 4	{**permit** | **deny** } *protocol* [**dest-option** | **dest-option-type** {*doh-number* | *doh-type*}] [**dscp** *cp-value*] [**flow-label** *fl-value*] [**mobility** | **mobility-type** {*mh-number* | *mh-type*}] [**routing** | **routing-type** *routing-number*] [**fragments**] [**log** | **log-input**] [**sequence** *seqno*] **Example:** `Device(config-ipv6rb-acl)# permit 33 dest-option dscp af11`	Specifies the access control entries (ACEs) for the RBACL. You can use most of the commands and options allowed in extended named access list configuration mode, with the source and destination fields omitted. The following ACE keywords are not supported: • **reflect** • **evaluate** • **time-range**

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config-ipv6rb-acl)# end`	Exits IPv6 role-based ACL configuration mode and returns to privileged EXEC mode.

Manually Applying SGACL Policies

To manually apply SGACL policies, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts role-based permissions default** [**ipv4** | **ipv6**] *sgacl-name1* [*sgacl-name2* [*sgacl-name3* ...]]] **Example:** `Device(config)# cts role-based permissions default MYDEFAULTSGACL`	Specifies the default SGACL. The default policies are applied when no explicit policy exists between the source and destination security groups.
Step 4	**cts role-based permissions from** {*source-sgt* | **unknown**} **to** {*dest-sgt* | **unknown**} [**ipv4** | **ipv6**] *sgacl-name1* [*sgacl-name2* [*sgacl-name3* ...]]] **Example:** `Device(config)# cts role-based permissions from 3 to 5 SRB3 SRB5`	Specifies the SGACLs to be applied for an SGT and a DGT. Values for *source-sgt* and *dest-sgt* range from 1 to 65533. By default, SGACLs are considered to be IPv4. • **from**: Specifies the source SGT. • **to**: Specifies the destination security group. • **unknown**: SGACL applies to packets where the security group (source or destination) cannot be determined. **Note** An SGACL policy downloaded dynamically from the ACS will override conflicting manual policies, if any.
Step 5	**end** **Example:**	Exits global configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	

Displaying SGACL Policies

After configuring the Cisco TrustSec device credentials and AAA, you can verify the Cisco TrustSec SGACL policies that are downloaded from the authentication server or configured manually. Cisco TrustSec downloads the SGACL policies when it learns of a new SGT Exchange Protocol (SXP) through authentication and authorization on an interface, from SXP, or from manual IP address to SGT mapping.

By using or omitting keywords, you can display all or part of the permissions matrix:

- If the **from** keyword is omitted, a column from the permissions matrix is displayed.
- If the **to** keyword is omitted, a row from the permissions matrix is displayed.
- If the **from** and **to** keywords are omitted, the entire permissions matrix is displayed.
- If the **from** and **to** keywords are specified, a single cell from the permissions matrix is displayed, and the **details** keyword is available. When **details** is entered, the ACEs of the SGACL of the single cell are displayed.

To display the contents of the SGACL policies' permissions matrix, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**show cts role-based permissions default** [**ipv4** | **ipv6** | **details**] **Example:** `Device# show cts role-based permissions default MYDEFAULTSGACL`	Displays the list of SGACL, of the default policy.
Step 3	**show cts role-based permissions from** {*source-sgt* | **unknown**} **to** {*dest-sgt* | **unknown**}] [**ipv4** | **ipv6** | **details**] **Example:** `Device# show cts role-based permissions from 3`	Specifies the SGACLs to be applied for an SGT and a DGT. Values for *source-sgt* and *dest-sgt* range from 1 to 65533. By default, SGACLs are considered to be IPv4. • **from**: Specifies the source SGT. • **to**: Specifies the destination security group. • **unknown**: SGACL applies to packets where the security group (source or destination) cannot be determined. **Note**

	Command or Action	Purpose
		An SGACL policy downloaded dynamically from the ACS will override conflicting manual policies, if any.
Step 4	**exit** **Example:** `Device# exit`	Exits privileged EXEC mode.

Refreshing the Downloaded SGACL Policies

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**cts refresh policy** {**peer** [*peer-id*] | **sgt** [*sgt_number* | **default** | **unknown**]} **Example:** `Device# cts refresh policy peer my_cisco_ise`	Performs an immediate refresh of the SGACL policies from the authentication server. • If a *peer-id* is specified, only the policies related to the specified peer connection are refreshed. To refresh all the peer policies, press **Enter** without specifying an ID. • If an SGT number is specified, only the policies related to that SGT are refreshed. To refresh all the SGT policies, press **Enter** without specifying an SGT number. Select **default** to refresh the default policy. Select **unknown** to refresh an unknown policy.
Step 3	**exit** **Example:** `Device# exit`	Exits privileged EXEC mode.

Configuration Examples for Security Group ACL Policies

The following sections provide examples of various SGACL policy configurations.

Example: Enabling SGACL Policy Enforcement Globally

The following example shows how to enable SGACL policy enforcement globally:

```
Device> enable
Device# configure terminal
Device(config)# cts role-based enforcement
```

Example: Enabling SGACL Policy Enforcement Per Interface

The following example shows how to enable SGACL policy enforcement per interface:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet 1/1
Device(config-if)# cts role-based enforcement
Device(config-if)# end
```

Example: Enabling SGACL Policy Enforcement on VLANs

The following example shows how to enable SGACL policy enforcement on VLANs:

```
Device> enable
Device# configure terminal
Device(config)# cts role-based enforcement vlan-list 31-35,41
Device(config)# exit
```

Example: Configuring SGACL Monitor Mode

The following example shows how to configure SGACL monitor mode:

```
Device> enable
Device# configure terminal
Device(config)# cts role-based monitor enable
Device(config)# cts role-based permissions from 2 to 3 ipv4
Device# show cts role-based permissions from 2 to 3 ipv4

IPv4 Role-based permissions from group 2:sgt2 to group 3:sgt3 (monitored):
        denytcpudpicmp-10
        Deny IP-00

Device# show cts role-based permissions from 2 to 3 ipv4 details

IPv4 Role-based permissions from group 2:sgt2 to group 3:sgt3 (monitored):
        denytcpudpicmp-10
        Deny IP-00
Details:
Role-based IP access list denytcpudpicmp-10 (downloaded)
        10 deny tcp
        20 deny udp
        30 deny icmp
Role-based IP access list Permit IP-00 (downloaded)
        10 permit ip
```

```
Device# show cts role-based counters ipv4

Role-based IPv4 counters
From     To     SW-Denied  HW-Denied  SW-Permitt  HW_Permitt  SW-Monitor  HW-Monitor
*         *     0          0          8           18962       0           0
2         3     0          0          0           0           0           341057
```

Example: Manually Configuring SGACL Policies

The following example shows how to manually configure SGACL policies:

```
Device> enable
Device# configure terminal
Device(config)# ip access role allow_webtraff
Device(config-rb-acl)# 10 permit tcp dst eq 80
Device(config-rb-acl)# 20 permit tcp dst eq 443
Device(config-rb-acl)# 30 permit icmp
Device(config-rb-acl)# 40 deny ip
Device(config-rb-acl)# exit
Device(config)# cts role-based permissions from 55 to 66 allow_webtraff

Device# show ip access allow_webtraff

Role-based IP access list allow_webtraff
    10 permit tcp dst eq www
    20 permit tcp dst eq 443
    30 permit icmp
    40 deny ip

Device# show cts role-based permissions from 2 to 5

Role-based permissions from group 2 to group 5:
srb2
srb5
```

Example: Manually Applying SGACLs

The following example shows how to manually apply SGACL policies:

```
Device> enable
Device# configure terminal
Device(config)# cts role-based permissions default MYDEFAULTSGACL
Device(config)# cts role-based permissions from 3 to 5 SRB3 SRB5
Device(config)# exit
```

Example: Displaying SGACL Policies

This example shows how to display the content of the SGACL policies permissions matrix for traffic sourced from security group 3:

```
Device> enable
Device# show cts role-based permissions from 3

Role-based permissions from group 3 to group 5:
```

```
        SRB3
        SRB5
Role-based permissions from group 3 to group 7:
        SRB4
```

CHAPTER 120

Configuring SGT Exchange Protocol

You can use the SGT Exchange Protocol (SXP) to propagate the Security Group Tags (SGTs) across network devices that do not have hardware support for Cisco Group-Based Policy. This module describes how to configure Cisco Group-Based Policy SXP on switches in your network.

Cisco Group-Based Policy builds secure networks by establishing domains of trusted network devices. Each device in the domain is authenticated by its peers. Communication on the links between devices in the domain is secured with a combination of encryption, message integrity check, and data-path replay protection mechanisms.

The Security Group Tag (SGT) Exchange Protocol (SXP) is one of several protocols that supports CTS and is referred to in this document as Cisco Group-Based Policy SXP. Cisco Group-Based Policy SXP is a control protocol for propagating IP-to-SGT binding information across network devices that do not have the capability to tag packets. Cisco Group-Based Policy SXP passes IP to SGT bindings from authentication points to upstream devices in the network. This process allows security services on switches, routers, or firewalls to learn identity information from access devices.

Prerequisites for SGT Exchange Protocol

The Cisco SGT Exchange Protocol (SXP) network needs to be established before implementing SXP. This network has the following prerequisites:

- To use the Cisco Group-Based Policy functionality on your existing router, ensure that you have purchased a Cisco Group-Based Policy security license. If the router is being ordered and needs the Cisco Group-Based Policy functionality, ensure that this license is pre-installed on your router before it is shipped to you
- Cisco Group-Based Policy SXP software must run on all network devices.
- Connectivity should exist between all network devices.

Restrictions for SGT Exchange Protocol

- Cisco Group-Based Policy Exchange Protocol is not supported on logical interfaces; supported only on physical interfaces.
- When the Dynamic Host Control Protocol (DHCP) snooping is enabled, Cisco Group-Based Policy enforcement for DHCP packets are passed by enforcement polices.
- Modifying a peer list under an SXP group is not supported when the peer connection configuration is present.
- Modifying an export list or import list under the speaker or listener export-import group is not allowed when an SXP connection configuration is present for any of the peers in the group. To modify the configuration under the export-import group, the corresponding peer SXP connection configuration must be removed. You can also shut down SXP by using the **no cts sxp enable** command.
- One peer cannot be configured under multiple export-import groups in the same direction, that is, a peer can be a part of the speaker export-import group as well as the listener export-import group but cannot be a part of a second speaker or listener group at the same time.
- Global export-import group configuration and per peer export-import group configuration are mutually exclusive.

Information About SGT Exchange Protocol

This section provides information about SGT Exchange Protocol.

SGT Exchange Protocol Overview

The Security Group Tag (SGT) Exchange Protocol (SXP) is one of several protocols that supports Cisco Group-Based Policy. SXP is a control protocol for propagating IP-to-SGT binding information across network devices that do not have the capability to tag packets. Cisco Group-Based Policy filters packets at the egress interface. During endpoint authentication, a host accessing the Cisco Group-Based Policy domain (the endpoint IP address) is associated with an SGT at the access device through Dynamic Host Control Protocol (DHCP) snooping and IP device tracking. The access device transmits that association or binding through SXP to Cisco Group-Based Policy hardware-capable egress devices. These devices maintain a table of source IP-to-SGT bindings. Packets are filtered on the egress interface by Cisco Group-Based Policy hardware-capable devices by applying security group access control lists (SGACLs). SXP passes IP-to-SGT bindings from authentication points to upstream devices in the network. This process allows security services on switches, routers, or firewalls to learn identity information from access devices.

SGTs can be assigned through any of the following Endpoint Admission Control (EAC) access methods:

- 802.1X port-based authentication
- MAC Authentication Bypass (MAB)
- Web Authentication

SXP uses TCP as the transport protocol, and the TCP port 64999 for connection initiation. SXP uses Message Digest 5 (MD5) and TCP Authentication Option (TCP-AO) for authentication and integrity check. It has two defined roles—speaker (initiator) and listener (receiver).

Security Group Tagging

Security Group Tag is a unique 16 bit tag that is assigned to a unique role. It represents the privilege of the source user, device, or entity and is tagged at the ingress of the Cisco Group-Based Policy domain. SXP uses the device and user credentials acquired during authentication for classifying packets by security groups (SGs) as they enter a network. This packet classification is maintained by tagging packets on the ingress to the Cisco Group-Based Policy network so that they can be identified for the purpose of applying security and other policy criteria along the data path. The Security Group Tag (SGT) allows the network to enforce the access control policy by enabling the endpoint device to act upon the SGT to filter traffic. Static port Identification is used to lookup the SGT value for a particular endpoint connected to a port.

SGT Assignment

The Security Group Tag (SGT) of a packet can be assigned at the port level when the packet comes tagged on a Cisco Group-Based Policy link, or when a single endpoint authenticates on a port. SGT of an incoming packet is determined in the following ways:

- When a packet that is tagged with an SGT comes on a trusted port, the tag in the packet is considered as the source SGT.
- When a packet is tagged with an SGT, but comes on an untrusted port, the packet is ignored and the source SGT is set as configured on the port.
- When a packet does not have an SGT, the source SGT is set as configured on the port.

The following methods of assigning SGTs are supported:

- IPM (dot1x, MAB, and Web Authentication)
- VLAN-to-SGT mapping is a low priority classification method where IP addresses used within the VLAN are learned through IP device tracking. The learned IP addresses are assigned to the static SGT.
- SXP (SGT Exchange Protocol) Listener
- IP SGT
- Subnet SGT
- Port SGT
- Caching SGT

SXP Version 5

The deployment of VRFs is dependent on SXP connections and IP-SGT mappings. With an increase in the number of VRFs, an increase in SXP connections along with IP-SGT mappings is required. To improve this dependency, SXP Version 5 has been designed to export and import SXP mappings between specified SXP peers. SXP Version 5 can export IP-SGT bindings under various user-defined VRFs over a single connection, unlike SXP Version 4, which can export only the connection VRF IP-SGT bindings over a single connection.

- SXP Version 5 exports certain mappings on the SXP speaker side based on the binding source type or VRF.
- SXP Version 5 imports certain mappings on the SXP listener side into the specified VRF.

Based on your configuration, which of the VRF-associated IP-SGT binding should be exported to the remote peer device is decided. If an SXP connection is created between two devices that support SXP Version 5, the SXP connection operates in SXP Version 5 mode. If a device at either end of the SXP connection supports a lower version of SXP, the SXP connection operates at the lowest of the supported versions.

You can configure a VRF or list of VRF tables on which IP-SGT binding should be exported to peer devices, by using the **cts sxp** global configuration command.

How to Configure SGT Exchange Protocol

This section describes how to configure SGT Exchange Protocol.

Configuring a Device SGT Manually

In a normal Cisco Group-Based Policy operation, the authentication server assigns an SGT to the device for packets originating from the device. You can manually configure an SGT to be used if the authentication server is not accessible, but an authentication server-assigned SGT will take precedence over a manually-assigned SGT.

To manually configure an SGT on the device, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**cts sgt** *tag* **Example:** `Device(config)# cts sgt tag`	Configures the SGT for packets sent from the device. The tag argument is in decimal format. The range is 1 to 65533.
Step 3	**exit** **Example:** `Device(config)# exit`	Exits configuration mode.

Configuring an SXP Peer Connection

You must configure the SXP peer connection on both of the devices. One device is the speaker and the other is the listener, or you can also set both speaker and listener in both the devices. When using password protection, make sure to use the same password on both ends.

Note If a default SXP source IP address is not configured and you do not configure an SXP source address in the connection, the Cisco Group-Based Policy software derives the SXP source IP address from existing local IP addresses. The SXP source address might be different for each TCP connection initiated from the device.

To configure an SXP peer connection, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts sxp connection peer** *peer-ipv4-addr*[**source** *src-ipv4-addr*] **password** {**default | none**} **mode** {**local | peer**} {**speaker | listener**} {**vrf** *vrf-name*} **Example:** `Device(config)# cts sxp connection peer 10.10.1.1 password default mode local listener`	Configures the SXP address connection. The optional **source** keyword specifies the IPv4 address of the source device. If no address is specified, the connection will use the default source address, if configured, or the address of the port. The **password** keyword specifies the password that SXP will use for the connection using the following options: • **default**—Use the default SXP password you configured using the **cts sxp default password** command. • **none**—Do not use a password. The mode keyword specifies the role of the remote peer device: • **local**—The specified mode refers to the local device. • **peer**—The specified mode refers to the peer device. • **speaker**—Default. Specifies that the device is the speaker in the connection. • **listener**—Specifies that the device is the listener in the connection.

	Command or Action	Purpose
		The optional **vrf** keyword specifies the VRF to the peer. The default is the default VRF.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode
Step 5	**show cts sxp connections** **Example:** `Device# show cts sxp connections`	(Optional) Displays the SXP connection information.

Configuring the Default SXP Password

By default, SXP uses no password when setting up connections.

To configure a default SXP password, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts sxp default password** [**0** | **6** | **7**] *password* **Example:** `Device(config)# cts sxp default password 0 hello`	Configures the SXP default password. You can enter either a clear text password (using the **0** or no option) or an encrypted password (using the **6** or **7** option). The maximum password length is 32 characters.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode

Configuring the Default SXP Source IP Address

SXP uses the default source IP address for all new TCP connections where a source IP address is not specified. There is no effect on existing TCP connections when you configure the default SXP source IP address.

To configure a default SXP source IP address, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts sxp default source-ip** *src-ip-addr* **Example:** `Device(config)# cts sxp default source-ip 10.0.1.2`	Configures the SXP default source IP address.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Changing the SXP Reconciliation Period

After a peer terminates an SXP connection, an internal hold-down timer starts. If the peer reconnects before the internal hold-down timer expires, the SXP reconciliation period timer starts. While the SXP reconciliation period timer is active, the Cisco Group-Based Policy software retains the SGT mapping entries learned from the previous connection and removes invalid entries. The default value is 120 seconds (2 minutes). Setting the SXP reconciliation period to 0 seconds disables the timer and causes all entries from the previous connection to be removed.

To change the SXP reconciliation period, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**cts sxp reconciliation period** *seconds* **Example:** `Device(config)# cts sxp reconciliation period 360`	Changes the SXP reconciliation timer. The default value is 120 seconds (2 minutes). The range is from 0 to 64000.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Changing the SXP Retry Period

The SXP retry period determines how often the Cisco Group-Based Policy software retries an SXP connection. When an SXP connection is not successfully set up, the Cisco Group-Based Policy software makes a new attempt to set up the connection after the SXP retry period timer expires. The default value is 120 seconds. Setting the SXP retry period to 0 seconds disables the timer and retries are not attempted.

To change the SXP retry period, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts sxp retry period** *seconds* **Example:** `Device(config)# cts sxp retry period 360`	Changes the SXP retry timer. The default value is 120 seconds (2 minutes). The range is from 0 to 64000.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Creating Syslogs to Capture Changes of IP Address-to-SGT Mapping Learned Through SXP

When the **cts sxp log binding-changes** command is configured in global configuration mode, SXP syslogs (sev 5 syslog) are generated whenever a change to IP address to SGT binding occurs (add, delete, change).

These changes are learned and propagated on the SXP connection. The default is **no cts sxp log binding-changes**.

To enable logging of binding changes, perform the following task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts sxp log binding-changes** **Example:** `Device(config)# cts sxp log binding-changes`	Enables logging for IP to SGT binding changes.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring an SXP Export List

To configure an SXP export list, perform this task.

Note Export-list configurations cannot be removed if they are associated with an SXP group. To remove it, you must first disable the SXP connection.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**cts sxp export-list** *export_list_name* **Example:** `Device(config)# cts sxp export-list export_list_1`	Configures an SXP export list, and enters export-list configuration mode.
Step 4	**binding-source-type** {**all** | **caching** | **cli** | **l3if** | **lisp-local-host** | **lisp-remote-host** | **local** | **omp** | **vlan**} **Example:** `Device(config-export-list)# binding-source-type all`	(Optional) Configures the bindings of the corresponding source type that are to be exported to the peer. • **all**: Exports all bindings. • **caching**: Exports cached bindings to a peer. • **cli**: Exports CLI bindings to a peer. • **l3if**: Exports L3IF bindings to a peer. • **lisp-local-host**: Exports LISP local bindings to a peer. • **lisp-remote-host**: Exports LISP remote bindings to a peer. • **local**: Exports local bindings to a peer. • **omp**: Exports OMP bindings to a peer. • **vlan**: Exports VLAN bindings to a peer.
Step 5	**vrf** {*instance_name* | **Default-vrf** | **all**} **Example:** `Device(config-export-list)# vrf all`	(Optional) Configures a VPN routing and forwarding instance. • *instance_name*: Specifies a VPN routing and forwarding instance name. • **Default-vrf**: Exports default VRF bindings. • **all**: Exports all IP-SGT bindings. **Note** **vrf all** and **vrf** *instance_name* configurations are mutually exclusive.
Step 6	**end** **Example:** `Device(config-export-list)# end`	Exits export list configuration mode, and returns to privileged EXEC mode

Configuring an SXP Import List

To configure an SXP import list, perform this task:

Note Import-list configurations cannot be removed if they are associated with an SXP group. To remove an import-list configuration, you must first disable the corresponding SXP connection.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts sxp import-list** *import_list_name* **Example:** `Device(config)# cts sxp import-list import_list_1`	Configures an SXP import list, and enters import list configuration mode.
Step 4	**vlan-list** **Example:** `Device(config-import-list)# vlan-list`	(Optional) Configures import VRF based on the VLAN in the received binding update. **Note** If there is no VRF mapping in the device for a VLAN received in the update, the bindings that are received are added to the default VRF table.
Step 5	**vrf** {*instance_name* | **Default-vrf**}} **Example:** `Device(config-import-list)# vrf vrf_1`	(Optional) Configures the VRF used to import the bindings. • *instance_name*: Specifies a VPN routing and forwarding instance name. • **Default-vrf**: Configures the default VPN routing and forwarding instance. **Note** **vrf** *instance_name* and **vlan-list** configuration are mutually exclusive.
Step 6	**end** **Example:** `Device(config-import-list)# end`	Exits export list configuration mode, and returns to privileged EXEC mode

Configuring an SXP Export-Import Group

The export-import groups are defined as either speaker or listener groups that control the export or import of SXP bindings for a group.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts sxp export-import-group** {**listener** | **speaker**} {**global** | *list_name*} **Example:** `Device(config)# cts sxp export-import-group listener group_1`	Configures an SXP export-import group, and enters export-import-group configuration mode. • **global**: Configures either an SXP listener global import group or an SXP speaker global export group. Global speaker or listener export-import group is applied to all the SXP connections configured in the device. • *list_name*: Specifies the default VPN routing and forwarding instance name.
Step 4	**import-list** *list_name* **Example:** `Device(config-export-import-group)# import-list import_1`	(Optional) Specifies the import list name to be applied to the export-import group. An empty import list or export list cannot be attached to a listener or speaker export-import group respectively.
Step 5	**export-list** *list_name* **Example:** `Device(config-export-import-group)# export-list export_1`	(Optional) Specifies the export list name to be applied to the export-import group. An empty import list or export list cannot be attached to a listener or speaker export-import group respectively.
Step 6	**peer** *address_name* **Example:** `Device(config-export-import-group)# peer 1.1.1.1 2.2.2.2`	(Optional) Configures a list of peers to be applied to the export-import group. A maximum of eight peers can be configured.
Step 7	**end** **Example:**	Exits export-import-group configuration mode, and returns to privileged EXEC mode

	Command or Action	Purpose
	`Device(config-export-import-group)# end`	

Configuration Examples for SGT Exchange Protocol

The following sections show configuration examples of SGT Exchange Protocol:

Example: Enabling Cisco Group-Based Policy SXP and an SXP Peer Connection

The following example shows how to enable SXP and configure an SXP peer connection between device A, the speaker, and device B, the listener:

```
Device# configure terminal
Device(config)# cts sxp enable
Device(config)# cts sxp default password Cisco123
Device(config)# cts sxp default source-ip 10.10.1.1
Device(config)# cts sxp connection peer 10.20.2.2 password default mode local speaker
```

The following example shows how to configure the SXP peer connection between device B, the listener, and device A, the speaker:

```
Device# configure terminal
Device(config)# cts sxp enable
Device(config)# cts sxp default password Cisco123
Device(config)# cts sxp default source-ip 10.20.2.2
Device(config)# cts sxp connection peer 10.10.1.1 password default mode local listener
```

Example: Configuring the Default SXP Password and Source IP Address

The following example shows how to configure a default SXP password and source IP address:

```
Device# configure terminal
Device(config)# cts sxp default password Cisco123
Device(config)# cts sxp default source-ip 10.20.2.2
Device(config)# end
```

Verifying SGT Exchange Protocol Connections

To view SXP connections, perform this task:

Command	Purpose
show cts sxp connections	Displays detailed information about the SXP status and connections.
show cts sxp connections [brief]	Displays brief information about the SXP status and connections.

Command	Purpose
show cts sxp export-list	Displays the list of VRFs associated with a specific export list or all the export lists.
show cts sxp import-list	Displays the list of VRFs associated with a specific import list name or all the import lists.
show cts sxp export-import-group [detailed]	Displays the export list or import list applied with a specific export-import group along with the list of peers that are a part of this export-import group.

The following is a sample output from the **show cts sxp connections** command:

```
Device# show cts sxp connections

SXP                    : Enabled
Default Password       : Set
Default Source IP      : 10.10.1.1
Connection retry open period: 10 secs
Reconcile period       : 120 secs
Retry open timer is not running
----------------------------------------------
Peer IP                : 10.20.2.2
Source IP              : 10.10.1.1
Conn status            : On
Conn Version           : 2
Connection mode        : SXP Listener
Connection inst#       : 1
TCP conn fd            : 1
TCP conn password      : default SXP password
Duration since last state change: 0:00:21:25 (dd:hr:mm:sec)
Total num of SXP Connections = 1
```

The following is a sample output from the **show cts sxp connections brief** command:

```
Device# show cts sxp connections brief

SXP                    : Enabled
Default Password       : Set
Default Source IP      : Not Set
Connection retry open period: 120 secs
Reconcile period       : 120 secs
Retry open timer is not running
------------------------------------------------------------------------------
Peer_IP          Source_IP          Conn Status    Duration
------------------------------------------------------------------------------
10.1.3.1         10.1.3.2           On             6:00:09:13 (dd:hr:mm:sec)
Total num of SXP Connections = 1
```

The following is a sample output of the **show cts sxp export-list** command displaying the list of VRFs associated with a specific export list or all the export lists configured on the device:

```
Device# show cts sxp export-list export_list_1

   Export-list-name: export_list_1
   vrf red_vrf
   vrf blue_vrf

Device# show cts sxp export-list

   Export-list-name: export_list_1
```

```
       vrf red_vrf
       vrf blue_vrf
       vrf green_vrf
   Export-list-name: export_list_2
       vrf all
```

The following is a sample output of the **show cts sxp export-import-group** command displaying the export list or import list applied to a specific export-import group along with the list of peers that are a part of this export-import group. The **show cts sxp export-import-group** command also lists the details of all the export-import groups configured on the device. Use the **detailed** keyword to display the export list or import list contents along with the export list or import list name and the list of peers. The **global** keyword displays the details of only the global listener and speaker.

```
Device# show cts sxp export-import-group speaker group_1

   Export-import-group: group_1
   Export-list-name: export_list_1
   Peer-list: 1.1.1.1, 2.2.2.2, 3.3.3.3

Device# show cts sxp export-import-group listener

Global Listener export-import-group: Not configured

   Export-import-group: group_1
   Export-list-name:  export_list_1
   Peer-list: 1.1.1.1, 2.2.2.2, 3.3.3.3

   Export-import-group: group_2
   Import-list-name: import_list_1
   Peer-list: 4.4.4.4, 5.5.5.5, 6.6.6.6


Device# show cts sxp export-import-group speaker group_1 detailed

    Export-import-group: group_1
    Export-list-name: export_list_1
    Export-list-content:
       vrf Red_vrf
       vrf Blue_vrf
    Peer-list: 1.1.1.1, 2.2.2.2, 3.3.3.3

Device# show cts sxp export-import-group listener detailed

    Global Listener export-import-group: Not configured

    Export-import-group: group_1
    Import-list-name: import_list_1
    Import-list-content:
       vlan-list
    Peer-list: 1.1.1.1, 2.2.2.2, 3.3.3.3

Device# show cts sxp export-import-group global

   Global Listener export-import-list Name: group_1
   Global Speaker export-import-list Name: group_2
```

CHAPTER 121

Configuring Security Group Tag Mapping

Subnet to security group tag (SGT) mapping binds an SGT to all host addresses of a specified subnet. Once this mapping is implemented, Cisco TrustSec imposes the SGT on any incoming packet that has a source IP address which belongs to the specified subnet.

Restrictions for SGT Mapping

Restrictions for Subnet-to-SGT Mapping

- An IPv4 subnetwork with a /31 prefix cannot be expanded.
- Subnet host addresses cannot be bound to Security Group Tags (SGT)s when the **network-map** *bindings* bindings parameter is less than the total number of subnet hosts in the specified subnets, or when bindings is 0.
- IPv6 expansions and propagation only occurs when Security Exchange Protocol (SXP) speaker and listener are running SXPv3, or more recent versions.

Restriction for Default Route SGT Mapping

- Default route configuration is accepted only with the subnet **/0**. Entering only the host-ip without the subnet /0 displays the following message:

```
Device(config)#cts role-based sgt-map 0.0.0.0 sgt 1000
Default route configuration is not supported for host ip
```

Information About SGT Mapping

This section provides information about SGT mapping.

Overview of Subnet-to-SGT Mapping

Subnet-to-SGT mapping binds an SGT to all host addresses of a specified subnet. Cisco TrustSec imposes the SGT on an incoming packet when the packet's source IP address belongs to the specified subnet. The subnet and SGT are specified in the CLI with the **cts role-based sgt-map** *net_address/prefix* **sgt** *sgt_number* global configuration command. A single host may also be mapped with this command.

In IPv4 networks, Security Exchange Protocol (SXP)v3, and more recent versions, can receive and parse subnet *net_address/prefix* strings from SXPv3 peers. Earlier SXP versions convert the subnet prefix into its set of host bindings before exporting them to an SXP listener peer.

For example, the IPv4 subnet 192.0.2.0/24 is expanded as follows (only 3 bits for host addresses):

- Host addresses 198.0.2.1 to 198.0.2.7—tagged and propagated to SXP peer.
- Network and broadcast addresses 198.0.2.0 and 198.0.2.8—not tagged and not propagated.

To limit the number of subnet bindings SXPv3 can export, use the **cts sxp mapping network-map** global configuration command.

Subnet bindings are static, there is no learning of active hosts. They can be used locally for SGT imposition and SGACL enforcement. Packets tagged by subnet-to-SGT mapping can be propagated on Layer 2 or Layer 3 Cisco TrustSec links.

For IPv6 networks, SXPv3 cannot export subnet bindings to SXPv2 or SXPv1 peers.

Overview of VLAN-to-SGT Mapping

The VLAN-to-SGT mapping feature binds an SGT to packets from a specified VLAN. This simplifies the migration from legacy to Cisco TrustSec-capable networks as follows:

- Supports devices that are not Cisco TrustSec-capable but are VLAN-capable, such as, legacy switches, wireless controllers, access points, VPNs, etc.
- Provides backward compatibility for topologies where VLANs and VLAN ACLs segment the network, such as, server segmentation in data centers.

The VLAN-to-SGT binding is configured with the **cts role-based sgt-map vlan-list** global configuration command.

When a VLAN is assigned a gateway that is a switched virtual interface (SVI) on a Cisco TrustSec-capable switch, and IP Device Tracking is enabled on that switch, then Cisco TrustSec can create an IP-to-SGT binding for any active host on that VLAN mapped to the SVI subnet.

IP-SGT bindings for the active VLAN hosts are exported to SXP listeners. The bindings for each mapped VLAN are inserted into the IP-to-SGT table associated with the VRF the VLAN is mapped to by either its SVI or by the **cts role-based l2-vrf** command.

VLAN-to-SGT bindings have the lowest priority of all binding methods and are ignored when bindings from other sources are received, such as from SXP or CLI host configurations. Binding priorities are listing in the Binding Source Priorities section.

Binding Source Priorities

Cisco TrustSec resolves conflicts among IP-SGT binding sources with a strict priority scheme. For example, an SGT may be applied to an interface with the **policy** {**dynamic identity** *peer-name* | **static sgt** *tag*} Cisco Trustsec Manual interface mode command (Identity Port Mapping). The current priority enforcement order, from lowest (1) to highest (7), is as follows:

1. VLAN: Bindings learned from snooped ARP packets on a VLAN that has VLAN-SGT mapping configured.
2. CLI: Address bindings configured using the IP-SGT form of the cts role-based sgt-map global configuration command.
3. SXP: Bindings learned from SXP peers.
4. IP_ARP: Bindings learned when tagged ARP packets are received on a CTS capable link.
5. LOCAL: Bindings of authenticated hosts which are learned via EPM and device tracking. This type of binding also include individual hosts that are learned via ARP snooping on L2 [I]PM configured ports.
6. INTERNAL: Bindings between locally configured IP addresses and the device own SGT.

Note If the source IP address matches multiple subnet prefixes with different assigned SGTs, then the longest prefix SGT takes precedence unless priority differs.

Default Route SGT

Default Route Security Group Tag (SGT) assigns an SGT number to default routes.

Default Route is that route which does not match a specified route and therefore is the route to the last resort destination. Default routes are used to direct packets addressed to networks not explicitly listed in the routing table.

How to Configure SGT Mapping

This section describes how to configure SGT mapping.

Configuring a Device SGT Manually

In normal Cisco TrustSec operation, the authentication server assigns an SGT to the device for packets originating from the device. You can manually configure an SGT to be used if the authentication server is not accessible, but an authentication server-assigned SGT will take precedence over a manually-assigned SGT.

To manually configure an SGT on the device, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device# enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts sgt** *tag* **Example:** `Device(config)# cts sgt 1234`	Enables SXP for Cisco TrustSec.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode

Configuring Subnet-to-SGT Mapping

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts sxp mapping network-map** *bindings* **Example:** `Device(config)# cts sxp mapping network-map 10000`	• Configures the Subnet to SGT Mapping host count constraint. The bindings argument specifies the maximum number of subnet IP hosts that can be bound to SGTs and exported to the SXP listener. • *bindings*—(0 to 65,535) default is 0 (no expansions performed)
Step 4	**cts role-based sgt-map** *ipv4_address/prefix* **sgt** *number* **Example:** `Device(config)# cts role-based sgt-map 10.10.10.10/29 sgt 1234`	(IPv4) Specifies a subnet in CIDR notation. • Use the *no* form of the command to unconfigure the Subnet to SGT mapping. The number of bindings specified in Step 2 should match or exceed the number of host addresses in the subnet (excluding network and broadcast addresses). The sgt

	Command or Action	Purpose
		number keyword specifies the Security Group Tag to be bound to every host address in the specified subnet. • *ipv4_address*—Specifies the IPv4 network address in dotted decimal notation. • *prefix*—(0 to 30) Specifies the number of bits in the network address. • **sgt** *number*—(0–65,535) Specifies the Security Group Tag (SGT) number.
Step 5	**cts role-based sgt-map** *ipv6_address::prefix* **sgt** *number* **Example:** `Device(config)# cts role-based sgt-map 2020::/64 sgt 1234`	(IPv6) Specifies a subnet in colon hexadecimal notation. Use the *no* form of the command to unconfigure the Subnet to SGT mapping. The number of bindings specified in Step 2 should match or exceed the number of host addresses in the subnet (excluding network and broadcast addresses). The sgt number keyword specifies the Security Group Tag to be bound to every host address in the specified subnet. • *ipv6_address*—Specifies IPv6 network address in colon hexadecimal notation. • *prefix*—(0 to128) Specifies the number of bits in the network address. • **sgt** *number*—(0–65,535) Specifies the Security Group Tag (SGT) number.
Step 6	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode..

Configuring VLAN-to-SGT Mapping

Task Flow for Configuring VLAN-SGT Mapping on a Cisco TrustSec device.

- Create a VLAN on the device with the same VLAN_ID of the incoming VLAN.
- Create an SVI for the VLAN on the device to be the default gateway for the endpoint clients.
- Configure the device to apply an SGT to the VLAN traffic.
- Enable IP Device tracking on the device.
- Attach a device tracking policy to a VLAN.

Note In a multi-switch network, SISF-based device tracking provides the capability to distribute binding table entries between switches running the feature. This assumes that binding entries are created on the switches where the host appears on an access port, and no entry is created for a host that appears over a trunk port. To achieve this in a multi-switch setup, we recommend that you configure another policy and attach it to the trunk port, as described in the *Configuring a Multi-Switch Network to Stop Creating Binding Entries from a Trunk Port* procedure, in the *Configuring SISF-Based Device Tracking* chapter of the *Security Configuration Guide*.

- Verify that VLAN-to-SGT mapping occurs on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan** *vlan_id* **Example:** `Device(config)# vlan 100`	Creates VLAN 100 on the TrustSec-capable gateway device and enters VLAN configuration mode.
Step 4	[**no**] **shutdown** **Example:** `Device(config-vlan)# no shutdown`	Provisions VLAN 100.
Step 5	**exit** **Example:** `Device(config-vlan)# exit`	Exits VLAN configuration mode and returns to global configuration mode.
Step 6	**interface** *type slot/port* **Example:** `Device(config)# interface vlan 100`	Specifies the interface type and enters interface configuration mode.
Step 7	**ip address** *slot/port* **Example:** `Device(config-if)# ip address 10.1.1.2 255.0.0.0`	Configures Switched Virtual Interface (SVI) for VLAN 100.

	Command or Action	Purpose
Step 8	[**no**] **shutdown** **Example:** `Device(config-if)# no shutdown`	Enables the SVI.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**cts role-based sgt-map vlan-list** *vlan_id* **sgt** *sgt_number* **Example:** `Device(config)# cts role-based sgt-map vlan-list 100 sgt 10`	Assigns the specified SGT to the specified VLAN.
Step 11	**device-tracking policy** *policy-name* **Example:** `Device(config)# device-tracking policy policy1`	Specifies the policy and enters device-tracking policy configuration mode.
Step 12	**tracking enable** **Example:** `Device(config-device-tracking)# tracking enable`	Overrides the default device tracking settings for the policy attribute.
Step 13	**exit** **Example:** `Device(config-device-tracking)# exit`	Exits device-tracking policy configuration mode and returns to global configuration mode.
Step 14	**vlan configuration** *vlan_id* **Example:** `Device(config)# vlan configuration 100`	Specifies the VLAN to which the device tracking policy will be attached, and enters the VLAN configuration mode.
Step 15	**device-tracking attach-policy** *policy-name* **Example:** `Device(config-vlan-config)# device-tracking attach-policy policy1`	Attaches a device tracking policy to the specified VLAN.
Step 16	**end** **Example:** `Device(config-vlan-config)# end`	Exits VLAN configuration mode and returns to privileged EXEC mode.
Step 17	**show cts role-based sgt-map** {*ipv4_netaddr* | *ipv4_netaddr/prefix* | *ipv6_netaddr* | *ipv6_netaddr/prefix* |**all** [**ipv4** |**ipv6**] |**host** { *ipv4__addr* |*ipv6_addr* } |**summary** [**ipv4** |**ipv6**]	(Optional) Displays the VLAN-to-SGT mappings.

	Command or Action	Purpose
	Example: `Device# show cts role-based sgt-map all`	
Step 18	**show device-tracking policy** *policy-name* **Example:** `Device# show device-tracking policy policy1`	(Optional) Displays the current policy attributes.

Emulating the Hardware Keystore

In cases where a hardware keystore is not present or is unusable, you can configure the switch to use a software emulation of the keystore. To configure the use of a software keystore, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts keystore emulate** **Example:** `Device(config)# cts keystore emulate`	Configures the switch to use a software emulation of the keystore instead of the hardware keystore.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits configuration mode.
Step 5	**show keystore** **Example:** `Device# show keystore`	Displays the status and contents of the keystore. The stored secrets are not displayed.

Configuring Default Route SGT

Before you begin

Ensure that you have already created a default route on the device using the **ip route 0.0.0.0** command. Otherwise, the default route (which comes with the Default Route SGT) gets an unknown destination and therefore the last resort destination will point to CPU.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts role-based sgt-map 0.0.0.0/0 sgt** *number* **Example:** `Device(config)# cts role-based sgt-map 0.0.0.0/0 sgt 3`	Specifies the SGT number for the default route. Valid values are from 0 to 65,519. **Note** • The **host_address/subnet** can be either IPv4 address (0.0.0.0/0) or IPv6 address (0:0::/0) • The default route configuration is accepted only with the subnet **/0**. Entering only the host-ip without the subnet /0 displays the following message: `Device(config)#cts role-based sgt-map 0.0.0.0 sgt 1000` `Default route configuration is not supported for host ip`
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.

Verifying SGT Mapping

The following sections show how to verify SGT mapping:

Verifying Subnet-to-SGT Mapping Configuration

To display Subnet-to-SGT Mapping configuration information, use one of the following show commands:

Command	Purpose
show cts sxp connections	Displays the SXP speaker and listener connections with their operational status.

Command	Purpose
show cts sxp sgt-map	Displays the IP to SGT bindings exported to the SXP listeners.
show running-config	Verifies that the subnet-to-SGT configurations commands are in the running configuration file.

Verifying VLAN-to-SGT Mapping

To display VLAN-to-SGT configuration information, use the following show commands:

Table 126:

Command	Purpose
show device-tracking policy	Displays the current policy attributes of the device tracking policy.
show cts role-based sgt-map	Displays IP address-to-SGT bindings.

Verifying Default Route SGT Configuration

Verify the Default Route SGT configuration:

```
device# show role-based sgt-map all
Active IPv4-SGT Bindings Information

IP Address              SGT     Source
============================================
0.0.0.0/0               3       CLI
11.0.0.0/8              11      CLI
11.0.0.10               1110    CLI
11.1.1.1                1111    CLI
21.0.0.2                212     CLI

IP-SGT Active Bindings Summary
============================================
Total number of CLI      bindings = 5
Total number of active   bindings = 5
```

Configuration Examples for SGT Mapping

The following sections show configuration examples of SGT mapping:

Example: Configuring a Device SGT Manually

```
Device# configure terminal
Device(config)# cts sgt 1234
Device(config)# exit
```

Example: Configuration for Subnet-to-SGT Mapping

The following example shows how to configure IPv4 Subnet-to-SGT Mapping between devices running SXPv3 (Device1 and Device2):

1. Configure SXP speaker/listener peering between devices.

```
Device1# configure terminal
Device1(config)# cts sxp enable
Device1(config)# cts sxp default source-ip 1.1.1.1
Device1(config)# cts sxp default password 1syzygy1
Device1(config)# cts sxp connection peer 2.2.2.2 password default mode local speaker
```

2. Configure Device2 as SXP listener of Device1.

```
Device2(config)# cts sxp enable
Device2(config)# cts sxp default source-ip 2.2.2.2
Device2(config)# cts sxp default password 1syzygy1
Device2(config)# cts sxp connection peer 1.1.1.1 password default mode local listener
```

3. On Device2, verify that the SXP connection is operating:

```
Device2# show cts sxp connections brief | include 1.1.1.1
          1.1.1.1                 2.2.2.2          On                 3:22:23:18
(dd:hr:mm:sec)
```

4. Configure the subnetworks to be expanded on Device1.

```
Device1(config)# cts sxp mapping network-map 10000
Device1(config)# cts role-based sgt-map 10.10.10.0/30 sgt 101
Device1(config)# cts role-based sgt-map 11.11.11.0/29 sgt 11111
Device1(config)# cts role-based sgt-map 192.168.1.0/28 sgt 65000
```

5. On Device2, verify the subnet-to-SGT expansion from Device1. There should be two expansions for the 10.10.10.0/30 subnetwork, six expansions for the 11.11.11.0/29 subnetwork, and 14 expansions for the 192.168.1.0/28 subnetwork.

```
Device2# show cts sxp sgt-map brief | include 101|11111|65000
            IPv4,SGT: <10.10.10.1 , 101>
            IPv4,SGT: <10.10.10.2 , 101>
            IPv4,SGT: <11.11.11.1 , 11111>
            IPv4,SGT: <11.11.11.2 , 11111>
            IPv4,SGT: <11.11.11.3 , 11111>
            IPv4,SGT: <11.11.11.4 , 11111>
            IPv4,SGT: <11.11.11.5 , 11111>
            IPv4,SGT: <11.11.11.6 , 11111>
            IPv4,SGT: <192.168.1.1 , 65000>
            IPv4,SGT: <192.168.1.2 , 65000>
            IPv4,SGT: <192.168.1.3 , 65000>
            IPv4,SGT: <192.168.1.4 , 65000>
            IPv4,SGT: <192.168.1.5 , 65000>
            IPv4,SGT: <192.168.1.6 , 65000>
            IPv4,SGT: <192.168.1.7 , 65000>
            IPv4,SGT: <192.168.1.8 , 65000>
            IPv4,SGT: <192.168.1.9 , 65000>
            IPv4,SGT: <192.168.1.10 , 65000>
            IPv4,SGT: <192.168.1.11 , 65000>
            IPv4,SGT: <192.168.1.12 , 65000>
            IPv4,SGT: <192.168.1.13 , 65000>
            IPv4,SGT: <192.168.1.14 , 65000>
```

6. Verify the expansion count on Device1:

```
Device1# show cts sxp sgt-map
            IP-SGT Mappings expanded:22
            There are no IP-SGT Mappings
```

7. Save the configurations on Device1 and Device2 and exit global configuration mode.

```
Device1(config)# copy running-config startup-config
Device1(config)# exit
Device2(config)# copy running-config startup-config
Device2(config)# exit
```

Example: Configuration for VLAN-to-SGT Mapping for a Single Host Over an Access Link

In the following example, a single host connects to VLAN 100 on an access device. A switched virtual interface on the TrustSec device is the default gateway for the VLAN 100 endpoint (IP Address 10.1.1.1). The TrustSec device imposes Security Group Tag (SGT) 10 on packets from VLAN 100.

1. Create VLAN 100 on an access device.

```
access_device# configure terminal
access_device(config)# vlan 100
access_device(config-vlan)# no shutdown
access_device(config-vlan)# exit
access_device(config)#
```

2. Configure the interface to the TrustSec device as an access link. Configurations for the endpoint access port are omitted in this example.

```
access_device(config)# interface gigabitEthernet 1/1
access_device(config-if)# switchport
access_device(config-if)# switchport mode access
access_device(config-if)# switchport access vlan 100
```

3. Create VLAN 100 on the TrustSec device.

```
TS_device(config)# vlan 100
TS_device(config-vlan)# no shutdown
TS_device(config-vlan)# end
TS_device#
```

4. Create an SVI as the gateway for incoming VLAN 100.

```
TS_device(config)# interface vlan 100
TS_device(config-if)# ip address 10.1.1.2 255.0.0.0
TS_device(config-if)# no shutdown
TS_device(config-if)# end
TS_device(config)#
```

5. Assign Security Group Tag (SGT) 10 to hosts on VLAN 100.

```
TS_device(config)# cts role-based sgt-map vlan 100 sgt 10
```

6. Enable IP Device Tracking on the TrustSec device. Verify that it is operating.

```
TS_device(config)# ip device tracking
TS_device# show ip device tracking all

IP Device Tracking = Enabled
```

```
IP Device Tracking Probe Count = 3
IP Device Tracking Probe Interval = 100
--------------------------------------------------------------------
  IP Address     MAC Address   Vlan   Interface         STATE
--------------------------------------------------------------------
Total number interfaces enabled: 1
Vlan100
```

7. (Optional) PING the default gateway from an endpoint (in this example, host IP Address 10.1.1.1). Verify that SGT 10 is being mapped to VLAN 100 hosts.

```
TS_device# show cts role-based sgt-map all

Active IP-SGT Bindings Information

IP Address         SGT         Source
============================================

10.1.1.1           10           VLAN

IP-SGT Active Bindings Summary
============================================
Total number of VLAN bindings = 1
Total number of CLI bindings = 0
Total number of active bindings = 1
```

Example: Emulating the Hardware Keystore

This example shows how to configure and verify the use of a software keystore:

```
Device# configure terminal
Device(config)# cts keystore emulate
Device(config)# exit
Device#show keystore
No hardware keystore present, using software emulation.
Keystore contains the following records (S=Simple Secret, P=PAC, R=RSA):
Index    Type     Name
-----    ----     ----
0         S       CTS-password
1         P       ECF05BB8DFAD854E8376DEA4EF6171CF
```

Example: Configuring Device Route SGT

```
Device# configure terminal
Device(config)# cts role-based sgt-map 0.0.0.0/0 sgt 3
Device(config)# exit
```

CHAPTER 122

Cisco TrustSec SGT Caching

The Cisco TrustSec SGT Caching feature enhances the ability of Cisco TrustSec to make Security Group Tag (SGT) transportability flexible. This feature identifies IP-SGT bindings, and caches the corresponding SGTs so that network packets are forwarded through all the network services for normal deep-packet inspection processing, and at the service egress point the packets are re-tagged with the appropriate SGT.

Only IPv4 SGT caching is supported. High availability is supported for SGT caching.

Restrictions for Cisco TrustSec SGT Caching

The global SGT caching configuration and the interface-specific ingress configuration are mutually exclusive. In the following scenarios, a warning message is displayed if you attempt to configure SGT caching both globally and on an interface:

- If an interface has ingress SGT caching enabled using the **cts role-based sgt-cache ingress** command in interface configuration mode, and a global configuration is attempted using the **cts role-based sgt-caching** command, a warning message is displayed, as shown in this example:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitEthernet 1/1
Device(config-if)# cts role-based sgt-cache ingress
Device(config-if)# exit
Device(config)# cts role-based sgt-caching

There is at least one interface that has ingress sgt caching configured. Please remove
 all interface ingress sgt caching configuration(s) before attempting global enable.
```

  This restriction specifically applies only to Layer 3-routed port interfaces. Also, the port must be a trusted port for SGT caching to work.

- Because SGT caching internally uses the NetFlow ternary content-addressable memory (TCAM) space, at any time on an interface, you can enable only either Flexible NetFlow or SGT caching in a given direction.

- If global configuration is enabled using the **cts role-based sgt-caching** command, and an interface configuration is attempted using the **cts role-based sgt-cache ingress** command in interface configuration mode, a warning message is displayed, as shown in this example:

```
Device> enable
Device# configure terminal
Device(config)# cts role-based sgt-caching
Device(config)# interface gigabitEthernet 1/1
Device(config-if)# cts role-based sgt-cache ingress

Note that ingress sgt caching is already active on this interface due to global
sgt-caching enable.
```

- IPv6 SGT caching is not supported.
- SGT caching cannot be performed for the link-local IPv6 source address.

  A link-local address is a network address that is valid only for communications within the network segment (link) or the broadcast domain that the host is connected to. Link-local addresses are not guaranteed to be unique beyond a single network segment. Therefore, devices do not forward packets with link-local addresses. Because they are not unique, SGT tags are not assigned for packets with source as link-local IPv6 address.
- SGT caching cannot coexist on the same port interface that has Application Visibility and Control (AVC), Wired Device AVC (WDAVC), Encrypted Traffic Analysis (ETTA,) or NetFlow/Flexible NetFlow features configured. An error message is displayed on the console if both SGT caching and one of these features are configured on the same interface.

  When SGT caching is enabled along with any of the above mentioned features, the following error message is displayed on the console: *SGT Caching cannot be configured. Remove the configuration.* However; the SGT Caching feature is displayed in the output of the **show running-config** command. You need to manually remove SGT caching and reconfigure it, after removing the feature that cannot co-exist with it.
-

Information About Cisco TrustSec SGT Caching

Identifying and Reapplying SGT Using SGT Caching

Cisco TrustSec uses Security Group Tag (SGT) caching to ensure that traffic that is tagged with SGT can also pass through services that are not aware of SGTs. Examples of services that cannot propagate SGTs are WAN acceleration or optimization, Intrusion Prevention Systems (IPSs), and upstream firewalls.

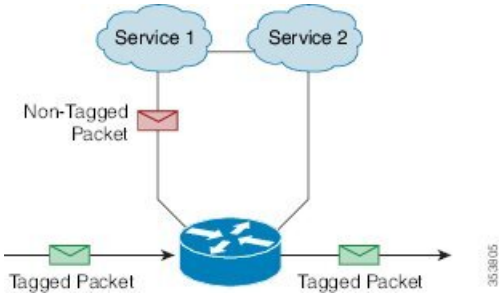
To configure SGACL caching on a VLAN, SGT caching must be enabled on the corresponding port and VLAN.

In one-arm mode (See the below figure), a packet tagged with SGT enters a device (where the tags are cached), and is redirected to a service. After that service is completed, the packet either returns to the device, or is redirected to another device. In such a scenario:

1. The Cisco TrustSec SGT Caching feature enables the device to identify the IP-SGT binding information from the incoming packet and caches this information.

2. The device redirects the packet to services that cannot propagate SGTs.
3. After the completion of the service, the packet returns to the device.
4. The appropriate SGT is reapplied to the packet at the service egress point.
5. Role-based enforcements are applied to the packet that has returned to the device from the service or services.
6. The packet with SGTs is forwarded to other Cisco TrustSec-capable devices downstream.

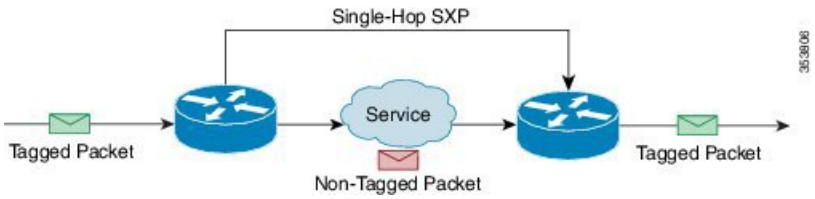
Figure 128: SGT Caching in One-Arm Mode



In certain instances, some services are deployed in a bump-in-the-wire topology (See the above figure). In such a scenario:

1. Packets that go through a service or services do not come back to the device.
2. Single-hop SGT Exchange Protocol (SXP) is used to identify and export the identified IP-SGT bindings.
3. The upstream device in the network identifies the IP-SGT bindings through SXP and reapplies the appropriate tags or uses them for SGT-based enforcement. During egress caching, the original pre-Network Address Translation (NAT) source IP address is cached as part of the identified IP-SGT binding information.
4. IP-SGT bindings that do not receive traffic for 300 seconds are removed from the cache.

Figure 129: SGT Caching in Bump-in-the-wire Topology



How to Configure Cisco TrustSec SGT Caching

This section describes how to configure SGT caching globally and on interfaces.

Configuring SGT Caching Globally

Before you begin

Before SGT caching is enabled, Security Exchange Protocol (SXP) must be established for information exchange.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts role-based sgt-caching** **Example:** `Device(config)# cts role-based sgt-caching`	Enables SGT caching in ingress direction for all interfaces.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring SGT Caching on an Interface

When an interface is configured to be on a Virtual Routing and Forwarding (VRF) network, the IP-SGT bindings identified on that interface are added under the specific VRF. (To view the bindings identified on a corresponding VRF, use the **show cts role-based sgt-map vrf** *vrf-name* **all** command.) SGT caching can also be configured per VRF.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot/port* **Example:** `Device(config)# interface gigabitethernet 1/1`	Configures an interface and enters interface configuration mode.
Step 4	**cts role-based sgt-cache** [**ingress** | **egress**] **Example:** `Device(config-if)# cts role-based sgt-cache ingress`	Configures SGT caching on a specific interface. • **ingress**: Enables SGT caching for traffic entering the specific interface (inbound traffic). • **ingress**: Enables SGT caching for traffic exiting the specific interface (outbound traffic).
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying Cisco TrustSec SGT Caching

Procedure

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **show cts**

Displays the Cisco TrustSec connections and the status of global SGT caching.

Example:

```
Device# show cts

Global Dot1x feature: Disabled
```

```
CTS device identity: ""
CTS caching support: disabled
CTS sgt-caching global: Enabled
Number of CTS interfaces in DOT1X mode:  0,    MANUAL mode: 0
Number of CTS interfaces in LAYER3 TrustSec mode: 0
Number of CTS interfaces in corresponding IFC state
  INIT            state:  0
  AUTHENTICATING  state:  0
  AUTHORIZING     state:  0
  SAP_NEGOTIATING state:  0
  OPEN            state:  0
  HELD            state:  0
  DISCONNECTING   state:  0
  INVALID         state:  0
CTS events statistics:
  authentication success: 0
  authentication reject : 0
  authentication failure: 0
  authentication logoff : 0
  authentication no resp: 0
  authorization success : 0
  authorization failure : 0
  sap success           : 0
  sap failure           : 0
  port auth failure     : 0
```

Step 3 **show cts interface**

Displays the Cisco TrustSec configuration statistics for an interface and SGT caching information with mode details (ingress or egress).

Example:

```
Device# show cts interface GigabitEthernet 1/1

Interface GigabitEthernet1/1
    CTS sgt-caching Ingress:  Enabled
    CTS sgt-caching Egress :  Disabled
    CTS is enabled, mode:     MANUAL
      Propagate SGT:          Enabled
      Static Ingress SGT Policy:
        Peer SGT:             200
        Peer SGT assignment:  Trusted

    L2-SGT Statistics
        Pkts In                     : 16298041
        Pkts (policy SGT assigned)  : 0
        Pkts Out                    : 5
        Pkts Drop (malformed packet): 0
        Pkts Drop (invalid SGT)     : 0
```

Step 4 **show cts interface brief**

Displays SGT caching information with mode details (ingress or egress) for all interfaces.

Example:

```
Device# show cts interface brief

Interface GigabitEthernet1/1
    CTS sgt-caching Ingress:  Enabled
```

```
    CTS sgt-caching Egress :  Disabled
    CTS is disabled

Interface GigabitEthernet1/1
    CTS sgt-caching Ingress:  Enabled
    CTS sgt-caching Egress :  Disabled
    CTS is enabled, mode:     MANUAL
      Propagate SGT:          Enabled
      Static Ingress SGT Policy:
        Peer SGT:             200
        Peer SGT assignment:  Trusted

Interface GigabitEthernet1/1
    CTS sgt-caching Ingress:  Enabled
    CTS sgt-caching Egress :  Disabled
    CTS is enabled, mode:     MANUAL
      Propagate SGT:          Enabled
      Static Ingress SGT Policy:
        Peer SGT:             0
        Peer SGT assignment:  Untrusted

Interface GigabitEthernet1/1
    CTS sgt-caching Ingress:  Enabled
    CTS sgt-caching Egress :  Disabled
    CTS is disabled

Interface Backplane-GigabitEthernet1/1
    CTS sgt-caching Ingress:  Enabled
    CTS sgt-caching Egress :  Disabled
    CTS is disabled

Interface RG-AR-IF-INPUT1
    CTS sgt-caching Ingress:  Enabled
    CTS sgt-caching Egress :  Disabled
    CTS is disabled
```

Step 5 **show cts role-based sgt-map all ipv4**

Displays all the SGT-IPv4 bindings.

Example:

```
Device# show cts role-based sgt-map all ipv4

Active IPv4-SGT Bindings Information

IP Address              SGT     Source
============================================
192.0.2.1                50      CACHED
192.0.2.2                50      CACHED
192.0.2.3                50      CACHED
192.0.2.4                50      CACHED
192.0.2.5                3900    INTERNAL
192.0.2.6                3900    INTERNAL
192.0.2.7                3900    INTERNAL

IP-SGT Active Bindings Summary
============================================
Total number of CACHED   bindings = 20
Total number of INTERNAL bindings = 3
Total number of active   bindings = 23
```

Step 6 **show cts role-based sgt-map vrf** *vrf-name* **all ipv4**

Displays all the SGT-IP bindings for a specific Virtual Routing and Forwarding (VRF) interface.

Example:

```
Device# show cts role-based sgt-map vrf vrf1 all ipv4

%IPv6 protocol is not enabled in VRF vrf1
Active IPv4-SGT Bindings Information

IP Address              SGT     Source
============================================
192.0.2.1                50      CACHED
192.0.2.2                2007    CACHED
192.0.2.3                50      CACHED
192.0.2.4                50      CACHED
```

Step 7 The SGT cache entry is removed after a port shutdown or SGT cache timeout.

Configuration Examples for Cisco TrustSec Caching

Example: Configuring SGT Caching Globally

The following example shows how to configure SGT caching globally:

```
Device> enable
Device# configure terminal
Device(config)# cts role-based sgt-caching
Device(config)# end
```

Example: Configuring SGT Caching for an Interface

The following example shows how to configure SGT caching for an interface:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitEthernet 1/1
Device(config-if)# cts role-based sgt-cache ingress
Device(config-if)# end
```

Example: Disabling SGT Caching on an Interface

The following example shows how to disable SGT caching on an interface and displays the status of SGT caching on the interface when caching is enabled globally, but disabled on the interface.

```
Device> enable
Device# configure terminal
Device(config)# cts role-based sgt-caching
Device(config)# interface gigabitEthernet 1/1
Device(config-if)# no cts role-based sgt-cache ingress
Device(config-if)# end
Device# show cts interface GigabitEthernet 1/1

Interface GigabitEthernet1/1
    CTS sgt-caching Ingress:  Disabled
    CTS sgt-caching Egress :  Disabled
    CTS is enabled, mode:     MANUAL
      Propagate SGT:          Enabled
      Static Ingress SGT Policy:
        Peer SGT:             200
        Peer SGT assignment:  Trusted

    L2-SGT Statistics
        Pkts In                     : 200890684
        Pkts (policy SGT assigned)  : 0
        Pkts Out                    : 14
        Pkts Drop (malformed packet): 0
        Pkts Drop (invalid SGT)     : 0
```

CHAPTER 123

IP-Prefix and SGT-Based SXP Filtering

The Security Group Tag (SGT) Exchange Protocol (SXP) is one of the several protocols that supports Cisco TrustSec. SXP is a control protocol for propagating IP-to-SGT binding information across network devices that do not have the capability to tag packets. SXP passes IP-to-SGT bindings from authentication points to upstream devices in a network. This process allows security services on switches, routers, or firewalls to learn user identity information from access devices.

The IP-Prefix and SGT-Based SXP Filtering feature allows IP-to-SGT bindings to be filtered, when they are exported or imported. This filtering can be done based on the IP prefix, SGT, or a combination of both.

Restrictions for IP-Prefix and Security Group Tag (SGT)-Based Security Exchange Protocol (SXP) Filtering

- No high availability support for the stateful synchronization of IP-Security Group Tag (SGT) bindings in an Security Exchange Protocol (SXP) database between active and standby devices.
- Filters applied to an existing connection will take effect only on the subsequent bindings that are exported or imported. The filters do not apply to any bindings that have been exported or imported prior to applying the filters.
- Virtual Routing and Forwarding (VRF)-specific filtering is not supported, and a filter specified for a peer IP is applicable across all VRFs on the device.
- SGT values in filter rules will be a list of single SGT numbers. SGT ranges are not supported.

Information About IP-Prefix and SGT-Based SXP Filtering

Overview

The IP-Prefix and SGT-Based SXP Filtering feature allows IP-to-SGT bindings to be filtered, when they are exported or imported. This filtering can be done based on the IP prefix, SGT, or a combination of both.

The Security Group Tag (SGT) Exchange Protocol (SXP) is one of the several protocols that supports Cisco TrustSec. SXP is a control protocol for propagating IP-to-SGT binding information across network devices that do not have the capability to tag packets. SXP passes IP-to-SGT bindings from authentication points to upstream devices in a network. This process allows security services on switches, routers, or firewalls to learn user identity information from access devices.

The IP-to-SGT filtering allow systems to selectively import or export only bindings of interest. In an SXP connection, a filter can be configured on a device that acts either as a speaker or a listener, based on the filtering that happens during the export or import of bindings.

In the case of bidirectional SXP connections, filters are applied in either of the directions, based on whether a speaker or listener filter is configured. If a peer is a part of both the speaker and the listener filter groups, then filtering is applied in both directions.

Filters can be applied either on a peer-to-peer basis or globally (applicable to all SXP connections). In both cases, the filter can be applied on the speaker or the listener.

Filter Rules

A filter that needs to be applied on a device is created with a set of filter rules. Each filter rule specifies the action or actions to be taken for bindings with specific SGT values and/or IP-prefix values. Each binding is matched against the values specified in the filter rules; if a match is found, the corresponding action specified in the filter rule is applied. An action that can be applied on a selected binding is either a permit or a deny action. When a filter is enabled on the speaker or listener during the export or import of IP-SGT bindings, the bindings are filtered based on the filter rules.

If a rule is not specified for a binding in a filter list, the catch-all rule that is configured in the filter-list is executed. In the absence of a catch-all rule, the corresponding binding is implicitly denied.

Types of SXP Filtering

IP-SGT bindings are filtered in one of the following ways:

- SGT-based filtering: Filters IP-SGT bindings in an SXP connection based on the SGT value.
- IP-prefix based filtering: Filters IP-SGT bindings in an SXP connection based on the IP-prefix value.
- SGT and IP-prefix based filtering: Filter IP-SGT bindings in an SXP connection based on the SGT value and IP-prefix value.

A filter rule is applied on each of the IP-SGT binding.

How to Configure IP-Prefix and SGT-Based SXP Filtering

This section describes how to configure IP-prefix and SGT-cased SXP filtering.

Configuring SXP Filter List

In this step, a filter list is created to hold a set of rules. These rules filter the IP-SGT bindings by allowing bindings that are permitted, and blocking bindings that are denied. Each rule can be based on an SGT, IP prefix, or a combination of both the SGT and IP prefix.

If a filter list does not have a rule that matches a specific IP-SGT binding, the binding is implicitly denied unless a default or catch-all ruled is defined.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal**	Enters global configuration mode.
Step 3	**cts sxp filter-list** *filter-name*	Configures a Cisco TrustSec filter list and enters filter-list configuration mode.
Step 4	*sequence-number* **permit ipv4** *ip-address/prefix* **deny sgt** *sgt-value*	Configures a filter list rule.
Step 5	**exit**	Exits filter-list configuration mode and returns to global configuration mode.
Step 6	**cts sxp filter-list** *filter-name*	Configures a Cisco TrustSec filter list and enters filter list configuration mode.
Step 7	[*sequence-number*] **deny sgt** *sgt-value* **permit ipv6** *ipv6-address/prefix*	Configures a filter list rule.
Step 8	**exit**	Exits filter-list configuration mode and returns to global configuration mode.
Step 9	**cts sxp filter-list** *filter-name*	Configures a Cisco TrustSec filter list and enters filter list configuration mode.
Step 10	[*sequence-number*] **permit ipv6** *ipv6-address/prefix* **permit** *sgt-value* **permit**	Configures a filter list rule.
Step 11	**end**	Exits filter-list configuration mode and returns to privileged EXEC mode.

Configuring SXP Filter Group

In this step, a set of peers are combined into a group, and a filter list is applied to the group. A filter-group can either be defined as a speaker group or listener group. To apply the same filter list to all speakers or all listeners, you can create a global speaker filter group or a global listener filter group.

Note Only one filter list can be attached to a filter group.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal**	Enters global configuration mode.
Step 3	**cts sxp filter-group listener** *listener-name*	Configures an SXP filter-group listener, and enters filter-group configuration mode.
Step 4	**filter** *filter-list-name*	Configures a filter list rule.
Step 5	**peer** *ipv4-address*	Configures the IP address of a peer.
Step 6	**exit**	Exits filter-group configuration mode and returns to global configuration mode.
Step 7	**cts sxp filter-group speaker** *speaker-name*	Configures a voice VLAN on a multiple VLAN access port.
Step 8	**filter** *filter-list-name*	Configures a filter list name.
Step 9	**peer** *ipv4-address*	Configures the IP address of a peer.
Step 10	**end**	Exits filter-group configuration mode and returns to privileged EXEC mode.

Configuring a Global Listener or Speaker Filter Group

When configuring a global listener and global speaker filter group, the filter is applied to across the box for all SXP connections that are in listener or speaker mode.

When adding a filter-list to a filter group the currently configured set of filter lists on the box is displayed as a help string.

Note The **peer** command is not available for the global listener and global speaker filter-group.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
		• Enter your password if prompted.
Step 2	**configure terminal**	Enters global configuration mode.
Step 3	**cts sxp filter-group listener global** *filter-list-name*	Configures a global listener filter group.
Step 4	**cts sxp filter-group speaker global** *filter-list-name*	Configures a global speaker filter group.
Step 5	**end**	Exits global configuration mode and returns to privileged EXEC mode.

Enabling SXP Filtering

After the SXP filter list and filter groups are configured, you must enable filtering.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal**	Enters global configuration mode.
Step 3	**cts sxp filter enable**	Configures a source template for the interface.
Step 4	**exit**	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**show cts sxp filter-list** *filter_name*	Displays the filter lists configured on the device along with the filter rules in each of the filter list.

Configuring the Default or Catch-All Rule

The default or catch-all rule is applied on IP-SGT bindings for which there was no match with any of the rules in the filter list. If a default rule is not specified, these IP-SGT bindings are denied.

Define the default or catch-all rule in the filter-list configuration mode of the corresponding filter list.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal**	Enters global configuration mode.
Step 3	**cts sxp filter-list** *filter-name*	Configures a Cisco TrustSec filter list and enters filter-list configuration mode.
Step 4	**permit ipv4** *ip-address/prefix*	Permits access if the conditions are matched.
Step 5	**deny ipv6** *ipv6-address/prefix*	Denies access if the conditions are matched.
Step 6	**permit sgt all**	Permits bindings corresponding to all SGTs.
Step 7	**end**	Exits filter-list configuration mode and returns to privileged EXEC mode.

Configuration Examples for IP-Prefix and SGT-Based SXP Filtering

The following sections show configuration examples of IP-prefix and SGT-based SXP filtering.

Example: Configuring an SXP Filter List

```
Device> enable
Device# configure terminal
Device(config)# cts sxp filter-list filter1
Device(config-filter-list)# permit ipv4 10.1.1.0/24 deny sgt 3 4
Device(config-filter-list)# exit
Device(config)# cts sxp filter-list filter2
Device(config-filter-list)# permit sgt all
Device(config-filter-list)# exit
Device(config)# cts sxp filter-list filter3
Device(config-filter-list)# deny ipv6 2001:db8::1/64 permit sgt 67
Device(config-filter-list)# end
```

Example: Configuring an SXP Filter Group

```
Device> enable
Device# configure terminal
Device(config)# cts sxp filter-group listener group1
Device(config-filter-group)# filter filter1
Device(config-filter-group)# peer 172.16.0.1 192.168.0.1
Device(config-filter-group)# exit
Device(config)# cts sxp filter-group listener global group2
Device(config)# end
```

Example: Enabling SXP Filtering

```
Device> enable
Device# configure terminal
Device(config)# cts sxp filter-enable
Device(config)# end
```

Example: Configuring the Default or Catch-All Rule

The following example shows how to create a default prefix rule that permits bindings corresponding to all IPv4 and IPv6 addresses:

```
Device(config)# cts sxp filter-list filter1
Device(config-filter-list)# permit ipv4 10.0.0.0/0
Device(config-filter-list)# deny ipv6 2001:db8::1/0
```

The following example shows how to create a default SGT rule that permits bindings corresponding to all SGTs:

```
Device(config)# cts sxp filter-list filter_1
Device(config-filter-list)# permit sgt all
```

Verifying IP-Prefix and SGT-Based SXP Filtering

To verify the configuration, use the following commands:

The **debug cts sxp filter events** command is used to log events related to the creation, removal, and update of filter-lists and filter-groups. This command is also used to capture events related to the matching actions in a filtering process.

```
Device# debug cts sxp filter events
```

The following sample output from the **show cts sxp filter-group speaker** command displays SXP speaker filter groups:

```
Device# show cts sxp filter-group speaker group1
   Filter-group: group1
   Filter-name: filter1
   Peer-list: 172.16.0.1 192.168.0.1
```

The following sample output from the **show cts sxp filter-group listener** command displays SXP speaker listener groups:

```
Device# show cts sxp filter-group listener

Global Listener Filter: Not configured
   Filter-group: group1
   Filter-name: filter1
   Peer-list: 172.16.0.1 192.168.0.1
   Filter-group: group2
   Filter-name: filter1
```

```
   Peer-list: 192.0.2.1, 198.51.100.1, 203.0.113.1
```

The following sample output from the **show cts sxp filter-group speaker detailed** command displays detailed information about SXP speaker filter groups:

```
Device# show cts sxp filter-group speaker group1 detailed

   Filter-group: group1
   Filter-name: filter1
   Filter-rules:
      10 deny sgt 30
      20 deny prefix 10.1.0.0/16
      30 permit sgt 60-100
   Peer-list: 172.16.0.1 192.168.0.1
```

The following sample output from the **show cts sxp filter-group** command displays information about all configured filter groups:

```
Device# show cts sxp filter-group

Global Listener Filter: Not configured

Global Speaker Filter: Not configured

Listener Group:
   Filter-group: group1
   Filter-name: filter1
   Peer-list: 172.16.0.1 192.168.0.1
   Filter-group: group2
   Filter-name: filter1
   Peer-list: 192.0.2.1, 198.51.100.1, 203.0.113.1

Speaker Group:
   Filter-group: group3
   Filter-name: filter1
   Peer-list: 172.16.0.1 192.168.0.13
   Filter-group: group2
   Filter-name: filter1
   Peer-list: 192.0.2.1, 198.51.100.1, 203.0.113.1
```

The following sample output from the **show sxp filter-group detailed** command displays detailed information about all configured SXP filter groups:

```
Device# show cts sxp filter-group detailed

Global Listener Filter: Configured
   Filter-name: global1
   Filter-rules:
      10 deny 192.168.0.13/32
      20 deny sgt 100-200

Global Speaker Filter: Configured
   Filter-name: global2
   Filter-rules:
      10 deny 192.168.0.13/32
      20 deny sgt 100-200

Listener Group:
   Filter-group: group1
```

```
   Filter-name: filter1
   Filter-rules:
      10 deny sgt 30
      20 deny prefix 172.16.0.0/16
      30 permit sgt 60-100
   Peer-list: 172.16.0.1, 192.168.0.13

   Filter-group: group2
   Filter-name: filter1
   Filter-rules:
      10 deny sgt 30
      20 deny prefix 172.16.0.0/16
      30 permit sgt 60-100
   Peer-list: 192.0.2.1, 198.51.100.1, 203.0.113.1

Speaker Group
   Filter-group: group3
   Filter-name: filter1
   Filter-rules:
      10 deny sgt 30
      20 deny prefix 172.16.0.0/16
      30 permit sgt 60-100
   Peer-list: 10.10.10.1, 172.16.0.1, 192.168.0.13

   Filter-group: group2
   Filter-name: filter1
   Filter-rules:
      10 deny sgt 30
      20 deny prefix 172.16.0.0/16
      30 permit sgt 60-100
   Peer-list: 192.0.2.1, 198.51.100.1, 203.0.113.1
```

Syslog Messages for SXP Filtering

Syslog messages for SXP filtering are generated to indicate the various events related to filtering.

Syslog Messages for Filter Rules

The maximum number of rules that can be configured in a single filter is 128. The following message is generated every time the number of filter rules that is configured in a single filter increases by 20% of the limit:

```
CTS SXP filter rules exceed %[ ] threshold. Reached count of [count] out of [max] in filter
 [filter-name].
```

The following message is generated when the number of rules configured in a single filter reaches 95% of the maximum number of rules allowed for a filter list:

```
CTS SXP filter rules exceed [ ] threshold. Reached count of [count] out of [max] in filter
 [filter-name].
```

The following message is generated when the number of rules configured in a single filter reaches the maximum number of allowed rules, and no more rules can be added.

```
Reached maximum filter rules. Could not add new rule in filter [filter-name]
```

Syslog Messages for Filter Lists

The maximum number of filter lists that can be configured is 256. The following message is generated every time the number of filter lists that is configured increases by 20% of this limit:

```
CTS SXP filter rules exceed %[ ] threshold. Reached count of [count] out of [max] in filter
 [filter-name].
```

The following message is generated when the number of filter lists that is configured reaches 95% of the maximum number of allowed filter lists:

```
CTS SXP filter rules exceed %[ ] threshold. Reached count of [count] out of [max]
```

The following message is generated when the number of filter lists that is configured reaches the maximum number of allowed filter lists, and no more filter lists can be added:

```
Reached maximum filter count. Could not add new filter
```

CHAPTER 124

Flexible NetFlow Export of Cisco TrustSec Fields

The Flexible NetFlow Export of Cisco TrustSec Fields feature supports the Cisco TrustSec fields in the Flexible NetFlow (FNF) flow record and helps to monitor, troubleshoot, and identify nonstandard behavior for Cisco TrustSec deployments.

This module describes the interaction between Cisco TrustSec and FNF and how to configure and export Cisco TrustSec fields in the NetFlow Version 9 flow records.

Restrictions for Flexible NetFlow Export of Cisco TrustSec Fields

- The security group tag (SGT) value that is exported in FNF records is zero in the following scenarios:
  - The corresponding packet is received with an SGT value of zero from a trusted interface.
  - The corresponding packet is received without an SGT.
  - The SGT is not found during the IP-SGT lookup. (The SGT is not found in the same packet because the packet is received without an SGT.)
  - When a flow record has SGT and Destination Group Tag (DGT) fields (or only either of the two), and if both these values are not applicable, a flow will still be created with zero values for SGT and DGT. The flow records are expected to include source and destination IP addresses, along with SGT and DGT fields.

Information About Flexible NetFlow Export of Cisco TrustSec Fields

Cisco TrustSec Fields in Flexible NetFlow

The Cisco TrustSec fields, source SGT and destination sSGT, in FNF flow records help administrators correlate the flow with identity information. It enables network engineers to gain a detailed understanding how customers use the network and application resources. This information can then be used to efficiently plan and allocate access and application resources, and to detect and resolve potential security and policy violations.

Cisco TrustSec fields are supported for ingress and egress FNF and for unicast and multicast traffic.

The following table lists NetFlow Version 9 enterprise-specific field types for Cisco TrustSec, which are used in FNF templates for the Cisco TrustSec source and destination SGTs.

Flow Field Type	Description
CTS_SRC_GROUP_TAG	Cisco TrustSec sourceSGT
CTS_DST_GROUP_TAG	Cisco TrustSec destination SGT

Cisco TrustSec fields are configured in addition to the existing match fields under the FNF flow record. The following configurations are used to add Cisco TrustSec flow objects to the FNF flow record as key or nonkey fields and to configure source and destination SGTs for a packet.

The **match flow cts {source | destination} group-tag** command is configured under the corresponding flow record to specify Cisco TrustSec fields as key fields. The key fields differentiate flows, with each flow having a unique set of values. A flow record requires at least one key field, before it can be used in a flow monitor. You can configure the **match** command to a source SGT, destination SGT or both, at the same time.

The flow record is then configured under the flow monitor, and the flow monitor is applied to an interface. To export the FNF data, a flow exporter needs to be configured and then added under the flow monitor.

How to Configure Flexible NetFlow Export of Cisco TrustSec Fields

The following sections provide information about the various tasks that comprise FNF export of Cisco TrustSec fields.

Configuring Cisco TrustSec Fields as Key Fields in Flow Record

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow record** *record-name* **Example:** `Device(config)# flow record cts-record-ipv4`	Creates a new FNF flow record, or modifies an existing FNF flow record, and enters Flexible NetFlow flow record configuration mode. • This command also allows you to modify an existing flow record.
Step 4	**match ipv4 protocol** **Example:** `Device(config-flow-record)# match ipv4 protocol`	(Optional) Configures the IPv4 protocol as a key field for a flow record.
Step 5	**match ipv4 source address** **Example:** `Device(config-flow-record)# match ipv4 source address`	(Optional) Configures the IPv4 source address as a key field for a flow record.
Step 6	**match ipv4 destination address** **Example:** `Device(config-flow-record)# match ipv4 destination address`	(Optional) Configures the IPv4 destination address as a key field for a flow record.
Step 7	**match transport source-port** **Example:** `Device(config-flow-record)# match transport source-port`	(Optional) Configures the transport source port as a key field for a flow record.
Step 8	**match transport destination-port** **Example:** `Device(config-flow-record)# match transport destination-port`	(Optional) Configures the transport destination port as a key field for a flow record.
Step 9	**match flow direction** **Example:**	(Optional) Configures the direction in which the flow is monitored as a key field.

	Command or Action	Purpose
	`Device(config-flow-record)# match flow direction`	
Step 10	**match flow cts** {**source** | **destination**} **group-tag** **Example:** `Device(config-flow-record)# match flow cts source group-tag` `Device(config-flow-record)# match flow cts destination group-tag`	Configures the Cisco TrustSec source group tag or destination group tag as a key field for the record in the FNF flow record. • Ingress: • In an incoming packet, if a header is present, SGT reflects the same value as the header. If no value is present, it will show zero. • The DGT value does not depend on the ingress port SGACL configuration. • Egress: • If either the **propagate-sgt** command, or Cisco TrustSec is disabled on the egress interface, SGT will be zero. • In an outgoing packet, if the SGACL configuration that corresponds to the SGT or DGT exists, DGT will be a numeral other than zero. • If SGACL is disabled on the egress port or VLAN, or if global SGACL enforcement is disabled, DGT will be zero.
Step 11	**end** **Example:** `Device(config-flow-record)# end`	Exits Flexible NetFlow flow record configuration mode and returns to privileged EXEC mode.

Configuring SGT Name Export in NetFlow

Each flow exporter supports only one destination. If you want to export the data to multiple destinations, you must configure multiple flow exporters and assign them to the flow monitor.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow exporter** *exporter-name* **Example:** `Device(config)# flow exporter EXPORTER-1`	Creates a flow exporter or modifies an existing flow exporter, and enters Flexible NetFlow flow exporter configuration mode.
Step 4	**destination** {*ip-address* | *hostname*} [**vrf** *vrf-name*] **Example:** `Device(config-flow-exporter)# destination 172.16.10.2`	Specifies the IP address or hostname of the destination system for the exporter.
Step 5	**option cts-sgt-table** [**timeout** *seconds*] **Example:** `Device(config-flow-exporter)# option cts-sgt-table timeout 1200`	Selects the SGT ID-to-name table option for the exporter. • This option allows FNF to export Cisco TrustSec environmental data tables that map SGTs to Security Group Names.
Step 6	**end** **Example:** `Device(config-flow-exporter)# end`	Exits Flexible NetFlow flow exporter configuration mode and returns to privileged EXEC mode.

Configuration Examples for Flexible NetFlow Export of Cisco TrustSec Fields

The following sections provide examples relating to the configuration of FNF export of Cisco TrustSec fields.

Example: Configuring Cisco TrustSec Fields as Key Fields in Flow Record

The following example shows how to configure the Cisco TrustSec flow objects as key fields in an IPv4 Flexible NetFlow flow record:

```
Device> enable
Device# configure terminal
Device(config)# flow record cts-record-ipv4
Device(config-flow-record)# match ipv4 protocol
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# match transport source-port
Device(config-flow-record)# match transport destination-port
Device(config-flow-record)# match flow direction
Device(config-flow-record)# match flow cts source group-tag
Device(config-flow-record)# match flow cts destination group-tag
Device(config-flow-record)# end
```

Example: Configuring SGT Name Export in NetFlow

The following example shows how to configure SGT Name Export in NetFlow.

```
Device> enable
Device# configure terminal
Device(config)# flow exporter EXPORTER-1
Device(config-flow-exporter)# destination 172.16.10.2
Device(config-flow-exporter)# option cts-sgt-table timeout 1200
Device(config-flow-exporter)# end
```

CHAPTER 125

TrustSec Security Group Name Download

The TrustSec Security Group Name Download feature enhances the Security Group Tag (SGT) policy that downloads to the network access device to include the SGT name in addition to the SGT number and Security Group Access Control List (SGACL) policy.

Layer 3 Logical Interface to SGT Mapping

The TrustSec Security Group Name Download feature is used to directly map SGTs to traffic of any of the following Layer 3 interfaces regardless of the underlying physical interface:

- Routed port
- SVI (VLAN interface)
- Layer3 subinterface of a Layer2 port
- Tunnel interface

The **cts role-based sgt-map interface** global configuration command to specify either a specific SGT number, or a Security Group Name (whose SGT association is dynamically acquired from a Cisco ISE or a Cisco ACS access server).

Configuring TrustSec Security Group Name Download

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cts role-based sgt-map interface** *type slot/port* [**security-group** *name* | **sgt** *number*] **Example:** `Device(config)# cts role-based sgt-map interface gigabitEthernet 1/1 sgt 77`	An SGT is imposed on ingress traffic to the specified interface. • **interface** *type slot*/*port*—Displays list of available interfaces. • **security-group** *name*— Security Group name to SGT pairings are configured on the Cisco ISE or Cisco ACS. • **sgt** *number*—(0 to 65,535). Specfies the Security Group Tag (SGT) number.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.
Step 5	**show cts role-based sgt-map all** **Example:** `Device# show cts role-based sgt-map all`	Verify that ingressing traffic is tagged with the specified SGT.

Example: TrustSec Security Group Name Download

The following example shows the SGT download configuration for the ingress interface:

```
Device# config terminal
Device(config)# cts role-based sgt-map interface gigabitEthernet 1/1 sgt 3
Device(config)# exit
```

The following example shows that ingressing traffic for the ingress interface is tagged appropriately:

```
Device# show cts role-based sgt-map all

IP Address              SGT     Source

============================================

15.1.1.15               4       INTERNAL

17.1.1.0/24             3       L3IF

21.1.1.2                4       INTERNAL

31.1.1.0/24             3       L3IF

31.1.1.2                4       INTERNAL

43.1.1.0/24             3       L3IF
```

```
49.1.1.0/24             3       L3IF

50.1.1.0/24             3       L3IF

50.1.1.2                4       INTERNAL

51.1.1.1                4       INTERNAL

52.1.1.0/24             3       L3IF

81.1.1.1                5       CLI

102.1.1.1               4       INTERNAL

105.1.1.1               3       L3IF

111.1.1.1               4       INTERNAL

IP-SGT Active Bindings Summary

============================================

Total number of CLI      bindings = 1

Total number of L3IF     bindings = 7

Total number of INTERNAL bindings = 7

Total number of active   bindings = 15
```

CHAPTER 126

Configuring SGT Inline Tagging

Restrictions for SGT Inline Tagging

- Cisco TrustSec manual configurations and 802.1x configurations can coexist only if Security Association Protocol is not configured.

Information About SGT Inline Tagging

Each security group in a Cisco TrustSec domain is assigned a unique 16 bit tag called the Security Group Tag (SGT). The SGT is a single label indicating the privileges of the source within the entire network. It is in turn propagated between network hops allowing any intermediary devices (switches, routers) to enforce polices based on the identity tag.

Cisco TrustSec-capable devices have built-in hardware capabilities than can send and receive packets with SGT embedded in the MAC (L2) layer. This feature is called Layer 2 (L2)-SGT Imposition. It allows ethernet interfaces on the device to be enabled for L2-SGT imposition so that the device can insert an SGT in the packet to be carried to its next hop ethernet neighbor. SGT-over-Ethernet is a method of hop-by-hop propagation of SGT embedded in clear-text (unencrypted) ethernet packets. The inline identity propagation is scalable, provides near line-rate performance and avoids control plane overhead.

The Cisco TrustSec with SGT Exchange Protocol V4 (SXPv4) feature supports Cisco TrustSec metadata-based L2-SGT. When a packet enters a Cisco TrustSec-enabled interface, the IP-SGT mapping database (with dynamic entries built by SXP and/or static entries built by configuration commands) is analyzed to learn the SGT corresponding to the source IP address of the packet, which is then inserted into the packet and carried throughout the network within the Cisco TrustSec header.

As the tag represents the group of the source, the tag is also referred to as the Source Group Tag (SGT). At the egress edge of the network, the group assigned to the packet's destination becomes known. At this point, access control can be applied. With Cisco TrustSec, access control policies are defined between the security

groups and are referred to as Security Group Access Control Lists (SGACL). From the view of any given packet, SGACL is simply being sourced from a security group and destined for another security group.

The SGT tag received in a packet from a trusted interface is propagated to the network, and is also be used for Identity firewall classification. When IPsec support is added, the received SGT tag is shared with IPSec for SGT tagging.

A network device at the ingress of Cisco TrustSec cloud needs to determine the SGT of the packet entering the Cisco TrustSec cloud so that it can tag the packet with that SGT when it forwards it into the Cisco TrustSec cloud. The SGT of a packet can be determined with these methods:

- SGT field on Cisco TrustSec header: If a packet is coming from a trusted peer device, it is assumed that the Cisco TrustSec header carries the correct SGT field. This situation applies to a network that is not the first network device in the Cisco TrustSec cloud for the packet.
- SGT lookup based on source IP address: In some cases, the administrator may manually configure a policy to decide the SGT of a packet based upon the source IP address. An IP address to SGT table can also be populated by the SXP protocol.

L2 Inline Tagging is supported for IPv6 multicast traffic with unicast source IPv6 addresses.

SGT Inline Tagging on a NAT Enabled Device

The following scenarios explain how SGT is determined for a packet that flows from a primary device, which has Network Address Translation (NAT) enabled on both ingress and egress ports, to a secondary device:

Note All ports that are used for the flow must have **CTS manual** and trusted configured on both devices.

- If inline tagging is enabled between both devices and SGT tag is not changed with CLI:

  In this case, on the primary device Cisco TrustSec is enforced on the SGT tag corresponding to the packet's source IP. The same SGT tag is tagged to the NAT IP. On the secondary device, Cisco TrustSec is enforced on the SGT tag corresponding to the packet's source IP also.

  For example, a packet is received on the primary device with a source IP 192.0.2.5 and SGT tag 133. Cisco TrustSec is enforced for the SGT tag 133 on the primary device. After NAT translation the packet's IP changes to 198.51.100.10 and tagged to the SGT tag 133. On the secondary device, the packet is received with IP address 198.51.100.10 and SGT tag 133. Cisco TrustSec is enforced with SGT tag 133 on the secondary device.

- If inline tagging is enabled between both devices and SGT tag is changed with CLI:

  In this case, on the primary device Cisco TrustSec is enforced on the SGT tag corresponding to the packet's source IP. The SGT tag is changed by CLI but the SGT tag corresponding to the packets's source IP is tagged to the packet's NAT IP. On the secondary device, Cisco TrustSec is enforced on the SGT tag corresponding to the packet's source IP also.

  For example, a packet is received on the primary device with a source IP 192.0.2.5 and SGT tag 133. Cisco TrustSec is enforced for the SGT tag 133 on the primary device. The SGT tag is changed to 200 with CLI. After NAT translation the packet’s IP changes to 198.51.100.10 but tagged to the SGT tag 133. On the secondary device, the packet is received with IP address 198.51.100.10 and SGT tag 133. Cisco TrustSec is enforced on the SGT tag 133 on the secondary device.

- If inline tagging is disabled (SGT is populated through SXP protocol on the secondary device) and SGT tag is changed with CLI:

  In this case, on the primary device Cisco TrustSec is enforced on the SGT tag corresponding to the packet's source IP. The SGT to Post Nat IP is defined through CLI and is learnt on the primary device. On the secondary device, Cisco TrustSec is enforced on the SGT tag corresponding to the NAT IP, if there is no direct Cisco TrustSec link between primary and secondary device and IP to SGT bindings are learnt through SXP in secondary device.

  For example, a packet is received on the primary device with a source IP 192.0.2.5 and SGT tag 133. After NAT translation the source IP changes to 198.51.100.10, for which the SGT is defined through CLI as 200. Cisco TrustSec is enforced for the SGT tag 133 on the primary device. On the secondary device, IP to SGT binding is received through SXP and Cisco TrustSec is enforced on the SGT tag 200 on the secondary device.

Configuring SGT Inline Tagging

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet port** **Example:** `Device(config)# interface gigabitethernet 1/1`	Configures the interface on which Cisco TrustSec SGT authorization and forwarding is enabled, and enters interface configuration mode.
Step 4	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Sets the trunking mode to access mode.

	Command or Action	Purpose
Step 5	**cts manual** **Example:** `Device(config-if)# cts manual`	Enables Cisco TrustSec SGT authorization and forwarding on the interface, and enters Cisco TrustSec manual interface configuration mode.
Step 6	**propagate sgt** **Example:** `Device(config-if-cts-manual)# propagate sgt`	Enables Cisco TrustSec SGT propagation on an interface. **Note** Use this command in situations where the peer device is not capable of receiving SGT over Ethernet packets (that is, when a peer device does not support Cisco Ethertype CMD 0x8909 frame format).
Step 7	**policy static sgt** *tag* [**trusted**] **Example:** `Device(config-if-cts-manual)# policy static sgt 77 trusted`	Configures a static SGT ingress policy on the interface and defines the trustworthiness of an SGT received on the interface. **Note** The **trusted** keyword indicates that the interface is trustworthy for Cisco TrustSec. The SGT value received in the Ethernet packet on this interface is trusted and will be used by the device for any SG-aware policy enforcement or for the purpose of egress-tagging.
Step 8	**exit** **Example:** `Device(config-if-cts-manual)# exit`	Exits Cisco TrustSec manual interface configuration mode and enters interface configuration mode.
Step 9	**dot1x pae authenticator** **Example:** `Device(config-if)# dot1x pae authenticator`	Enables 802.1x authentication on the port.
Step 10	**dot1x authenticator eap profile** *name* **Example:** `Device(config-if)# dot1x authenticator eap profile md5`	Specifies the Extensible Authentication Protocol (EAP) profile to use during 802.1x authentication.

	Command or Action	Purpose
Step 11	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.

Example: Configuring SGT Static Inline Tagging

This example shows how to enable an interface on the device for L2-SGT tagging or imposition and defines whether the interface is trusted for Cisco TrustSec

```
Device# configure terminal
Device(config)# interface gigabitethernet 1/1
Device(config-if)# cts manual
Device(config-if-cts-manual)# propagate sgt
Device(config-if-cts-manual)# policy static sgt 77 trusted
```

CHAPTER 127

Configuring Endpoint Admission Control

This module describes the Endpoint Admission Control (EAC) access methods for authentication and authorization in TrustSec networks.

Information About Endpoint Admission Control

In TrustSec networks, packets are filtered at the egress, not the ingress to the network. In TrustSec endpoint authentication, a host accessing the TrustSec domain (endpoint IP address) is associated with a Security Group Tag (SGT) at the access device through DHCP snooping and IP device tracking. The access device transmits that association (binding) through SXP-to-TrustSec hardware-capable egress devices, which maintain a continually updated table of Source IP to SGT bindings. Packets are filtered on egress by the TrustSec hardware-capable devices by applying security group ACLS (SGACLs).

Endpoint Admission Control (EAC) access methods for authentication and authorization can include the following:

- 802.1X port-based Authentication
- MAC Authentication Bypass (MAB)
- Web Authentication (WebAuth)

All port-based authentication can be enabled with the authentication command. Each access method must be configured individually per port. The flexible authentication sequence and failover features permit the administrator to specify the failover and fallback sequence when multiple authentication modes are configured and the active method fails. The 802.1X host mode determines how many endpoint hosts can be attached per 802.1X port.

Example: 802.1X Authentication Configuration

The following example shows the basic 802.1x configuration on a Gigabit Ethernet port:

```
Device> enable
Device# configure terminal
Device(config)# dot1x system-auth-control
Device(config)# interface GigabitEthernet1/1
Device(config-if)# authentication port-control auto
Device(config-if)# dot1x pae authenticator
```

Example: MAC Authentication Bypass Configuration

MAC Authentication Bypass (MAB) enables hosts or clients that are not 802.1X capable to join 802.1X-enabled networks. It is not required to enable 802.1X authentication prior to enabling MAB.

The following example is of a basic MAB configuration:

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet1/1
Device(config-if)# authentication port-control auto
Device(config-if)# mab
```

For additional information on configuring MAB authentication, see the configuration guide for your access device.

Example: Web Authentication Proxy Configuration

Web Authentication Proxy (WebAuth) allows the user to use a web browser to transmit their login credentials to the Cisco Secure ACS though a Cisco IOS web server on the access device. WebAuth can be enabled independently. It does not require 802.1X or MAB to be configured.

The following example shows a basic WebAuth configuration on a Gigabit Ethernet port:

```
Device(config)# ip http server
Device(config)# ip access-list extended POLICY
Device(config-ext-nacl)# permit udp any any eq bootps
Device(config-ext-nacl)# permit udp any any eq domain
Device(config)# ip admission name HTTP proxy http
Device(config)# fallback profile FALLBACK_PROFILE
Device(config-fallback-profile)# ip access-group POLICY in
Device(config-fallback-profile)# ip admission HTTP
Device(config)# interface GigabitEthernet1/1
Device(config-if)# authentication port-control auto
Device(config-if)# authentication fallback FALLBACK_PROFILE6500(config-if)#ip access-group
POLICY in
```

Example: Flexible Authentication Sequence and Failover Configuration

Flexible Authentication Sequence (FAS) allows the access port to be configured for 802.1X, MAB, and WebAuth authentication methods, specifying the fallback sequence if one or more of the authentication methods are not available. The default failover sequence is as follows:

- 802.1X port-based Authentication
- MAC Authentication Bypass
- Web Authentication

Layer 2 authentications always occur before Layer 3 authentications. That is, 802.1X and MAB must occur before WebAuth.

The following example specifies the authentication sequence as MAB, dot1X, and then WebAuth:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitEthernet 1/1
Device(config-if)# authentication order mab dot1x webauth
Device(config-if)# ^Z
```

For additional information on FAS, see Flexible Authentication Order, Priority, and Failed Authentication.

802.1X Host Modes

Four host classification modes can be configured per port:

- Single Host —Interface-based session with one MAC address
- Multi Host—Interface-based session with multiple MAC addresses per port
- Multi Domain—MAC + Domain (VLAN) session
- Multi Auth—MAC-based session with multiple MAC address per port

Pre-Authentication Open Access

The Pre-Authentication Open Access feature allows clients and devices to gain network access before port authentication is performed. This process is primarily required for the PXE boot scenario, where a device needs to access the network before PXE times out and download a bootable image that may contain a supplicant.

Example: DHCP Snooping and SGT Assignment

After the authentication process, authorization of the device occurs (for example, dynamic VLAN assignment, ACL programming, etc.). For TrustSec networks, a Security Group Tag (SGT) is assigned per the user

configuration in the Cisco ACS. The SGT is bound to traffic sent from that endpoint through DHCP snooping and the IP device tracking infrastructure.

The following example enables DHCP snooping and IP device tracking on an access device:

```
Device> enable
Device# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)# ip dhcp snooping
Device(config)# ip dhcp snooping vlan 10
Device(config)# no ip dhcp snooping information option
Device(config)#interface gigabitethernet 1/8
Device(config-if)#device-tracking
```

CHAPTER 128

Network Edge Access Topology

802.1x Supplicant and Authenticator Switches with Network Edge Access Topology

The 802.1x standard defines a client-server-based access control and authentication protocol that prevents unauthorized clients from connecting to a LAN through publicly accessible ports unless they are properly authenticated. The authentication server authenticates each client connected to a switch port before making available any services offered by the switch or the LAN. For more information about 802.1x, including configuration information, see Configuring IEEE 802.1x Port-Based Authentication.

The Network Edge Access Topology (NEAT) feature extends identity to areas outside the wiring closet. This allows any type of device to authenticate on the port. NEAT uses Client Information Signalling Protocol (CISP) to propagate Client MAC and VLAN information between supplicant and Authenticator. CISP and NEAT are supported only on L2 ports, not on L3 ports. You can configure NEAT on IE3x00 and ESS3300 switches.

- 802.1x switch supplicant: You can configure a switch to act as a supplicant to another switch by using the 802.1x supplicant feature. This configuration is helpful in a scenario, where, for example, a switch is outside a wiring closet and is connected to an upstream switch through a trunk port. A switch configured with the 802.1x switch supplicant feature authenticates with the upstream switch for secure connectivity. Once the supplicant switch authenticates successfully the port mode changes from access to trunk in an authenticator switch. In a supplicant switch you must manually configure the trunk when enabling CISP.
- If the access VLAN is configured on the authenticator switch, it becomes the native VLAN for the trunk port after successful authentication.

In the default state, when you connect a supplicant switch to an authenticator switch that has BPDU guard enabled, the authenticator port could be error-disabled if it receives a Spanning Tree Protocol (STP) bridge protocol data unit (BPDU) packets before the supplicant switch has authenticated. You can control traffic exiting the supplicant port during the authentication period. Entering the **dot1x supplicant controlled transient**

global configuration command temporarily blocks the supplicant port during authentication to ensure that the authenticator port does not shut down before authentication completes. If authentication fails, the supplicant port opens. Entering the **no dot1x supplicant controlled transient** global configuration command opens the supplicant port during the authentication period. This is the default behavior.

We strongly recommend using the **dot1x supplicant controlled transient**command on a supplicant switch when BPDU guard is enabled on the authenticator switch port with the **spanning-tree bpduguard enable** interface configuration command.

Note If you globally enable BPDU guard on the authenticator switch by using the **spanning-tree portfast bpduguard default** global configuration command, entering the **dot1x supplicant controlled transient** command on the Supplicant switch does not prevent the BPDU violation.

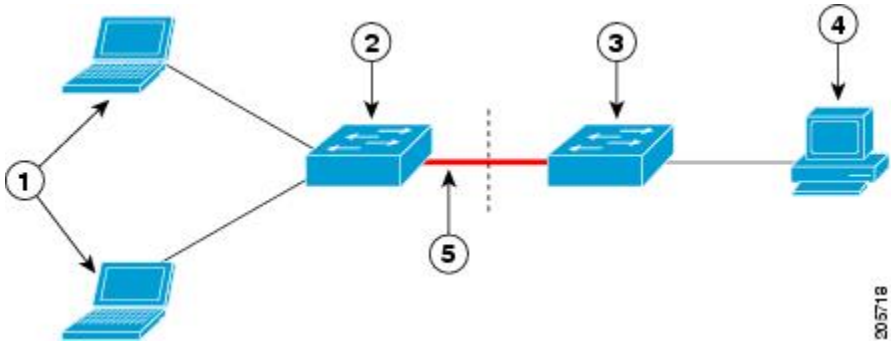
You can enable MDA or multiauth mode on the authenticator switch interface that connects to one more supplicant switches. Multihost mode is not supported on the authenticator switch interface.

When you reboot an authenticator switch with single-host mode enabled on the interface, the interface may move to err-disabled state before authentication. To recover from err-disabled state, flap the authenticator port to activate the interface again and initiate authentication.

Use the **dot1x supplicant force-multicast** global configuration command on the supplicant switch for NEAT to work in all host modes.

- Host Authorization: Ensures that only traffic from authorized hosts (connecting to the switch with supplicant) is allowed on the network. The switches use CISP to send the MAC addresses connecting to the supplicant switch to the authenticator switch.
- Auto enablement: Automatically enables trunk configuration on the authenticator switch, allowing user traffic from multiple VLANs coming from supplicant switches. Configure the cisco-av-pair as *device-traffic-class=switch* at the ISE. (You can configure this under the *group* or the *user* settings.)

Figure 130: Authenticator and Supplicant Switch Using CISP



1	Workstations (clients)
2	Supplicant switch (outside wiring closet)
3	Authenticator switch
4	Cisco ISE

5	Trunk port

Note The **switchport nonegotiate** command is not supported on supplicant and authenticator switches with NEAT. This command should not be configured at the supplicant side of the topology. If configured on the authenticator side, the internal macros will automatically remove this command from the port.

Guidelines and Limitations

The following are guidelines and limitations for configuring and using NEAT.

- A Radius server such as Cisco's Identity Server Engine (ISE) is required.
- CISP and NEAT are supported only on L2 ports, not on L3 ports.
- NEAT and 802.1x are not supported on EtherChannel ports.
- NEAT is not supported on dynamic ports.
- MACsec is supported with NEAT.
- NEAT can operate with PTP.
- MAB and NEAT are mutually exclusive. You cannot enable MAB when NEAT is enabled on an interface, and you should not enable NEAT when MAB is enabled on an interface.

Configuring an Authenticator Switch with NEAT

Configuring this feature requires that one switch outside a wiring closet is configured as a supplicant and is connected to an authenticator switch.

Note
- The *cisco-av-pairs* must be configured as *device-traffic-class=switch* on the ISE, which sets the interface as a trunk after the supplicant is successfully authenticated.

Beginning in privileged EXEC mode, follow these steps to configure a switch as an authenticator:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cisp enable** **Example:** `Device(config)# cisp enable`	Enables CISP.
Step 4	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 5	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Sets the port mode to **access**.
Step 6	**authentication port-control auto** **Example:** `Device(config-if)# authentication port-control auto`	Sets the port-authentication mode to auto.
Step 7	**dot1x pae authenticator** **Example:** `Device(config-if)# dot1x pae authenticator`	Configures the interface as a port access entity (PAE) authenticator.
Step 8	**spanning-tree portfast** **Example:** `Device(config-if)# spanning-tree portfast trunk`	Enables the interface to quickly transition to spanning-tree forwarding state for an interface which is a member of multiple VLANs. Use this command only when you are sure that the switch-to-switch connection is not part of a Layer2 loop.
Step 9	**end** **Example:**	Exits interface configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	

Configuring a Supplicant Switch with NEAT

Beginning in privileged EXEC mode, follow these steps to configure a switch as a supplicant:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cisp enable** **Example:** `Device(config)# cisp enable`	Enables CISP.
Step 4	**eap profile** *profile-name* **Example:** `Device(config)# eap profile CISP`	Creates an Extensible Authentication Protocol (EAP) profile and enters EAP profile configuration mode.
Step 5	**method** *type* **Example:** `Device(config-eap-profile)# method md5`	Specifies the EAP authentication method.
Step 6	**exit** **Example:** `Device(config-eap-profile)# exit`	Exits EAP profile configuration mode.

	Command or Action	Purpose
Step 7	**dot1x credentials** *profile* **Example:** `Device(config)# dot1x credentials test`	Creates 802.1x credentials profile. This must be attached to the port that is configured as supplicant.
Step 8	**username** *suppswitch* **Example:** `Device(config)# username suppswitch`	Creates a username.
Step 9	**password** *password* **Example:** `Device(config)# password myswitch`	Creates a password for the new username.
Step 10	**dot1x supplicant force-multicast** **Example:** `Device(config)# dot1x supplicant force-multicast`	Forces the switch to send only multicast EAPOL packets when it receives either unicast or multicast packets. This also allows NEAT to work on the supplicant switch in all host modes.
Step 11	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the port to be configured, and enters interface configuration mode.
Step 12	**switchport trunk encapsulation dot1q** **Example:** `Device(config-if)# switchport trunk encapsulation dot1q`	Sets the port to trunk mode.
Step 13	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Configures the interface as a VLAN trunk port.
Step 14	**dot1x pae supplicant** **Example:**	Configures the interface as a port access entity (PAE) supplicant.

	Command or Action	Purpose
	`Device(config-if)# dot1x pae supplicant`	
Step 15	**dot1x credentials** *profile-name* **Example:** `Device(config-if)# dot1x credentials test`	Attaches the 802.1x credentials profile to the interface.
Step 16	**dot1x supplicant eap profile** *profile-name* **Example:** `Device(config-if)# dot1x supplicant eap profile cisp`	Assigns the EAP-TLS profile to the 802.1X interface.
Step 17	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying Configuration

Use the following show commands to verify information about Client Information Signalling Protocol (CISP) and Network Edge Access Topology (NEAT) configuration:

- show cisp interface <interface name>
- show cisp clients
- show cisp summary
- show cisp registrations

Following is example output for **show cisp** commands. GigabitEthernet 1/1 is configured as Authenticator, and GigabitEthernet 1/2 is configured as Supplicant.

```
Auth# show cisp interface Gi1/2

CISP Status for interface Gi1/2
 -------------------------------------
Version: 1
Mode: Supplicant Peer
Mode: Authenticator
Supp State: Idle

Auth# show cisp clients

Authenticator Client Table:
 ----------------------
 MAC Address VLAN Interface
```

```
 --------------------------------
 0050.5695.4de8 1 Gi1/10
6c03.09e7.3947 1 Gi1/10
6c03.09e7.3954 11 Gi1/10
6c03.09e7.4485 1 Gi1/10
 9077.ee4a.8567 1 Gi1/10
e41f.7ba1.bbd4 1 Gi1/10

Supplicant Client Table:
------------------------
MAC Address VLAN Interface
--------------------------------
9077.ee4a.856b 11 Vl11
9077.ee4a.8572 1 Ap1/1
e41f.7bc7.2f03 1 Gi1/9

Auth# show cisp summary

CISP is running on the following interface(s):
---------------------------------------------
Gi1/2 (Authenticator)

Supp# show cisp summary

CISP is running on the following interface(s):
---------------------------------------------
Gi1/1 (Supplicant)

Auth# show cisp registrations

Interface(s) with CISP registered user(s):
------------------------------------------
Gi1/2
Auth Mgr (Authenticator)

Supp# show cisp registration

Interface(s) with CISP registered user(s):
------------------------------------------
Gi1/1
802.1x Sup (Supplicant)
```

Use the following debug commands to troubleshoot CISP and NEAT:

- debug access-session errors
- debug access-session event
- debug dot1x errors
- debug dot1x packets
- debug dot1x events

Configuration Example

Following is an example of Client Information Signalling Protocol (CISP) and Network Edge Access Topology (NEAT) configuration on the authenticator switch.

```
conf t
aaa new-model
```

```
cisp enable
radius server RADIUS_CWA
address ipv4 <ISE-IP> auth-port 1645 acct-port 1646
key <ISE KEY>
exit
aaa group server radius ISE
server name RADIUS_CWA
exit
aaa authentication dot1x default group radius
aaa authorization exec default group radius
aaa authorization network default group radius
aaa server radius dynamic-author
client <ISE-IP> server-key cisco123
dot1x system-auth-control
policy-map type control subscriber Policy_dot1x
event session-started match-all
10 class always do-until-failure
10 authenticate using dot1x
exit

interface <interface name>
switchport mode access
access-session closed
access-session port-control auto
dot1x pae authenticator
spanning-tree portfast
service-policy type control subscriber Policy_dot1x
exit
```

Following is an example of CISP and NEAT configuration on the supplicant switch.

```
conf t
cisp enable
eap profile CISP
method md5
exit
dot1x system-auth-control
dot1x supplicant controlled transient
dot1x credentials SWITCH
username <user configured in ISE>
password 0 <Password configured in ISE>
exit
interface <interface name>
switchport mode trunk
dot1x pae supplicant
dot1x credentials SWITCH
dot1x supplicant eap profile CISP
spanning-tree portfast trunk
exit
```

CHAPTER 129

Layer 2 Network Address Translation

Layer 2 Network Address Translation

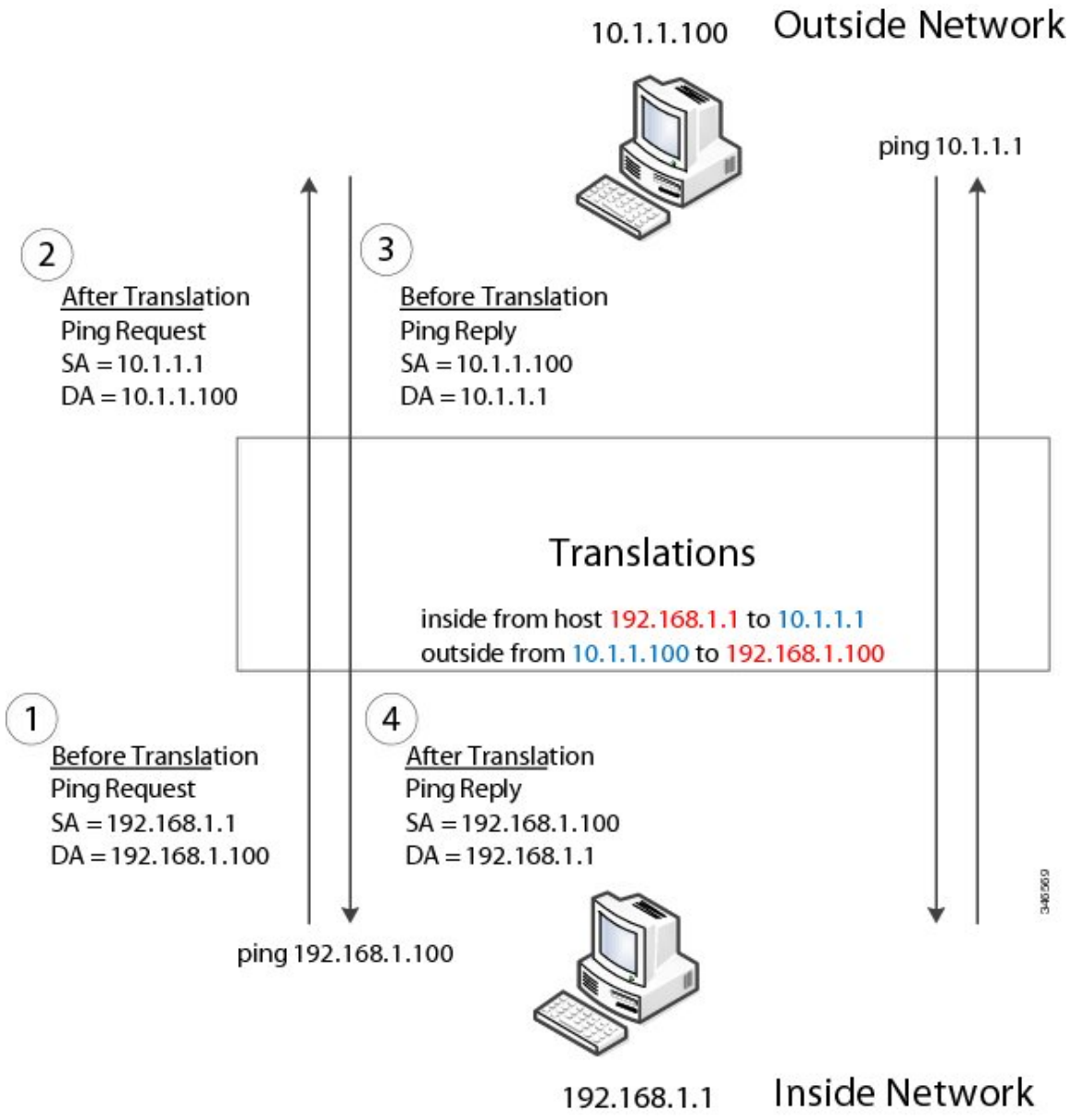
One-to-one Layer 2 NAT (Network Address Translation) is a service that allows the assignment of a unique public IP address to an existing private IP address (end device). The assignment enables the end device to communicate on both the private and public subnets. This service is configured in a NAT-enabled device and is the public “alias” of the IP address that is physically programmed on the end device. This is typically represented by a table in the NAT device.

Layer 2 NAT uses a table to translate IPv4 addresses both public-to-private, and private-to-public at line rate. Layer 2 NAT is a hardware-based implementation that provides the same high level of (bump-on-the-wire) wire-speed performance. This implementation also supports multiple VLANs through the NAT boundary for enhanced network segmentation.

In the following example, Layer 2 NAT translates addresses between sensors on a 192.168.1.x network and a line controller on a 10.1.1.x network.

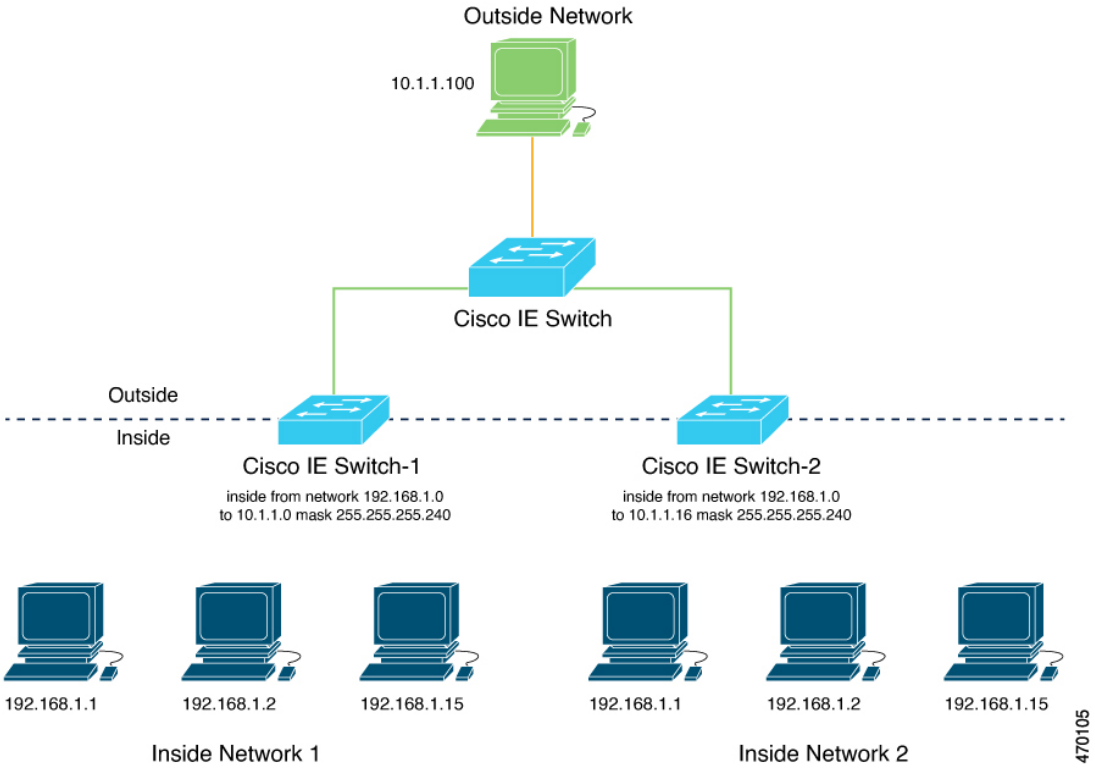
1. The 192.168.1.x network is the inside/internal IP address space and the 10.1.1.x network is the outside or external IP address space.
2. The sensor at 192.168.1.1 sends a ping request to the line controller by using an “inside” address, 192.168.1.100.
3. Before the packet leaves the internal network, Layer 2 NAT translates the source address (SA) to 10.1.1.1 and the destination address (DA) to 10.1.1.100.
4. The line controller sends a ping reply to 10.1.1.1.
5. When the packet is received on the internal network, Layer 2 NAT translates the source address to 192.168.1.100 and the destination address to 192.168.1.1.

Figure 131: Translating Addresses Between Networks



For large numbers of nodes, you can quickly enable translations for all devices in a subnet. In the scenario shown in the following figure, addresses from Inside Network 1 can be translated to outside addresses in the 10.1.1.0/28 subnet, and addresses from Inside Network 2 can be translated to outside addresses in the 10.1.1.16/28 subnet. All addresses in each subnet can be translated with one command. The benefit of using subnet-based translations saves in Layer L2 NAT rules. The switch has limits on the number of Layer 2 NAT rules. A rule with a subnet allows for multiple end devices to be translated with a single rule.

Figure 132: Inside-Outside Address Translation



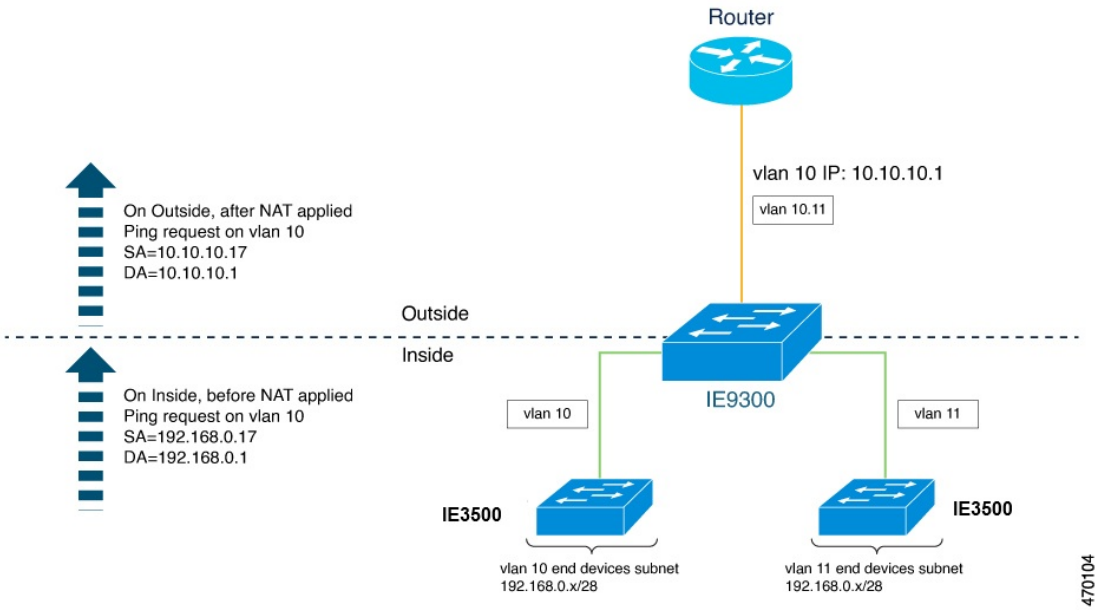
The following figure shows a switch at the aggregation layer forwarding Ethernet packets based on Layer 2 MAC Addresses. In this example, the router is the Layer 3 gateway for all subnets and VLANs.

The L2NAT instance definitions use the **network** command to define a translated row for multiple devices in the same subnet. In this the case, it's a /28 subnet with last byte in the IP address starting with 16 and ending with 31. The gateway for the VLAN is the router with last byte of the IP address ending with .1. An outside host translation is provided for the router. The **network** command in the Layer 2NAT definition translates a subnet's worth of host with a single command, saving on Layer 2 NAT translation records.

The Gi1/1 uplink interface has Layer 2NAT translation instances for vlan10 and vlan 11 subnets. Interfaces can support multiple Layer 2 NAT instance definitions.

The downstream switches are examples of access layer switches which do not perform L2NAT and rely on the upstream aggregation layer switch to do it.

Figure 133: NAT on the Switch



The following example shows the NAT configuration for the preceding diagram:

```
!
l2nat instance Subnet10-NAT
 instance-id 1
 permit all
 fixup all
 outside from host 10.10.10.1 to 192.168.0.1
 inside from network 192.168.0.0 to 10.10.10.16 mask 255.255.255.240
!
l2nat instance Subnet11-NAT
 instance-id 1
 permit all
 fixup all
 outside from host 10.10.11.1 to 192.168.0.1
 inside from network 192.168.0.0 to 10.10.11.16 mask 255.255.255.240
!
interface GigabitEthernet1/1
 switchport mode trunk
 l2nat Subnet10-NAT 10
 l2nat Subnet11-NAT 11
!
Interface vlan 1
  ip address 10.10.1.2
```

Guidelines and Limitations

The following list provides guidelines and limitations for using Layer 2 NAT with the Switch.

Note For scale information, see the section NAT Performance and Scalability, on page 1814 in this guide.

- Layer 2 NAT is supported on standalone switches.
- Layer 2 NAT is disabled by default; it becomes enabled when you configure it. See Configure Layer 2 NAT, on page 1814 in this guide.
- Layer 2 NAT applies only to unicast traffic. Untranslated unicast traffic, multicast traffic, and IGMP traffic are permitted.
- Layer 2 NAT is supported only on the uplink ports and available in both Network Essentials and Network Advantage licenses.
- Layer 2 NAT supports one-to-one mapping between external and internal IP addresses.
- Layer 2 NAT can be applied to uplink interfaces in access or trunk mode.
- Only IPv4 addresses for Layer 2 traffic can be translated.
- Supported subnet masks on inside network translation are /24, /25, /26, /27, /28, and /32 only.
- Outside translation rule supports only host translations.
- ARP does not work transparently across Layer 2 NAT; however, the switch changes the IP addresses embedded in the payload of IP packets for the protocols to work. Embedded IP addresses are not translated.
- Statistics for debugging include the following statistics: entries for each translation, translated total ingress and egress for each instance, and for each interface. Also included are ARP fixup stats and the number of translations entries allocated in hardware.
- Layer 2 NAT does not support one-to-many and many-to-one IP address mapping.
- Layer 2 NAT cannot save on public IP addresses because public-to-private is a 1:1 translation. It is not 1:N NAT.
- If you configure a translation for a Layer 2 NAT host, do not configure it as a DHCP client.
- When translating an inside address to an outside address using Layer 2 NAT, ensure that the translated IP address is not accessible in the global network.
- The management interface is behind the Layer 2 NAT function. Therefore this interface should not be on the private network VLAN. If it is on the private network VLAN, assign an inside address and configure an inside translation.
- Because Layer 2 NAT is designed to separate outside and inside addresses, we recommend that you do not configure addresses of the same subnet as both outside and inside addresses.
- Layer 2 NAT is only for Layer 2 traffic; do not use it for packets undergoing routing
- Layer 2 NAT does not translate packets destined for CPU and packets coming from CPU. Management traffic should be on a different VLAN from the private network VLAN.
- Layer 2 NAT counters are not based on ports. When the same Layer 2 NAT instance is applied to multiple interfaces, the corresponding Layer 2 NAT counters will be displayed for all those interfaces.

NAT Performance and Scalability

Layer 2 NAT translation and forwarding are performed in the hardware at line rate. The number of Layer 2NAT rules that are supported depends on the number of hardware entries that can be supported in hardware.

Scale depends on the number of inside/outside combinations. The following list provides scale examples.

- An instance with only inside rules can have a total of 128 translation rules.
- Multiple instances with one inside rule can have a total of 128 such instances applied to 128 different VLANs.
- Multiple instances with one inside rule and one outside rule can have a maximum of 64 instances.
- A single instance with one outside rule can have a maximum of 100 inside rules. The number of inside rules that can be supported reduces with increase in the outside rules.

Note We recommended that you use network translation rules to save on the number of rules.

Configure Layer 2 NAT

You must configure Layer 2 NAT instances that specify the address translations. Attach Layer 2 NAT instances to physical Ethernet interfaces, and configure which VLAN or VLANs the instances will be applied to. Layer 2 NAT instances can be configured from management interfaces (CLI/SNMP). You can view detailed statistics about the packets that are sent and received. See the section Verify the Configuration, on page 1817 in this guide.

To configure Layer 2 NAT, follow these steps. Refer to the examples in Basic Inside-to-Outside Communications: Example, on page 1818 and Duplicate IP Addresses Example, on page 1821 in this guide for more details.

Procedure

Step 1 Enter global configuration mode:

configure terminal

Step 2 Create a new Layer 2 NAT instance:

l2nat instance *instance_name* After creating an instance, you use this same command to enter the submode for that instance.

Step 3 Translate an inside address to an outside address:

inside from [*host* | *range* | *network*] *original ip* to *translated ip* [*mask*] *number* | *mask*

You can translate a single host address, a range of host addresses, or all the addresses in a subnet. Translate the source address for outbound traffic and the destination address for inbound traffic.

Step 4 Translate an outside address to an inside address:

outside from [*host* | *range* | *network*] *original ip* to *translated ip* [*mask*] *number* | *mask*

You can translate a single host address, a range of host addresses, or the addresses in a subnet. Translate the destination address for outbound traffic and the source address for inbound traffic.

Step 5 Exit config-l2nat mode:

exit

Step 6 Access interface configuration mode for the specified interface (uplink ports only on the IE 3400):

interface *interface-id*

Step 7 Apply the specified Layer 2 NAT instance to a VLAN or VLAN range. If this parameter is missing, the Layer 2 NAT instance applies to the native VLAN.

l2nat *instance_name* [*vlan* | *vlan_range*]

Step 8 Exit interface configuration mode:

end

Configure Layer 2 NAT support on Port Channel

Note Layer 2 NAT is supported on logical interface of port-channel but not on member interface.

The LACP is defined in IEEE 802.3ad and enables Cisco devices to manage port-channels between devices that conform to the IEEE 802.3ad protocol. LACP facilitates the automatic creation of port-channels by exchanging LACP packets between Ethernet ports.

By using LACP, the switch or switch stack learns the identity of partners capable of supporting LACP and the capabilities of each port. It then dynamically groups similarly configured ports into a single logical link (channel or aggregate port). Similarly configured ports are grouped based on hardware, administrative, and port parameter constraints. For example, LACP groups the ports with the same speed, duplex mode, native VLAN, VLAN range, and trunking status and type. After grouping the links into an port-channels, LACP adds the group to the spanning tree as a single device port.

LACP modes specify whether a port can send LACP packets or only receive LACP packets.

Active mode: Places a port into an active negotiating state in which the port starts negotiations with other ports by sending LACP packets.

Passive mode: Places a port into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation. This setting minimizes the transmission of LACP packets.

Both the active and passive LACP modes enable ports to negotiate with partner ports to a port-channel based on criteria such as port speed, and for Layer 2 EtherChannels, based on trunk state and VLAN numbers.

When you specify the maximum number of bundled LACP ports allowed in a port channel, the remaining ports in the port channel are designated as hot-standby ports. Beginning in privileged EXEC mode, follow these steps to configure the maximum number of LACP ports in a port-channel. This procedure is optional.

Procedure

Step 1 Enter global configuration mode:

device configure

Step 2 Create a new Layer 2 NAT instance called A-LC:

device # l2nat instance A-LC

Step 3 Translate A1's inside address to an outside address:

Device(config-l2nat)# inside from host 192.168.1.1 to 10.1.1.1

Step 4 Translate A2's inside address to an outside address:

Device(config-l2nat)# inside from host 192.168.1.2 to 10.1.1.2

Step 5 Translate A3's inside address to an outside address:

Device(config-l2nat)# inside from host 192.168.1.3 to 10.1.1.3

Step 6 Translate LC's outside address to an inside address:

Device(config-l2nat)# outside from host 10.1.1.200 to 192.168.1.250

Step 7 Exit config-l2nat mode:

Device(config-l2nat)# exit

Step 8 Access interface configuration mode for the port channel:

Device(config)# interface port-channel

Step 9 Apply this Layer 2 NAT instance to the native VLAN on this interface:

Device(config-if)#l2nat A-LC

Note
For tagged traffic on a trunk, add the VLAN number when attaching the instance to an interface as follows:
`l2nat instance vlan`

Step 10 Return to privileged EXEC mode:

Device# end

Verify the Configuration

Procedure

Perform the following commands to verify the Layer 2 NAT configuration.

Command	Purpose
show l2nat instance	Displays the configuration details for a specified Layer 2 NAT instance.
show l2nat interface	Displays the configuration details for Layer 2 NAT instances on one or more interfaces.
show l2nat statistics	Displays the Layer 2 NAT statistics for all interfaces.
show l2nat statistics interface	Displays the Layer 2 NAT statistics for a specified interface.
debug l2nat	Enables showing real-time Layer 2 NAT configuration details when the configuration is applied.
show platform hardware fed switch 1 fwd-asic resource tcam table pbr record 0 format 0 -	Displays the hardware entries.
-show platform hardware fed switch active fwd-asic resource tcam utilization | in PBR	Displays the hardware resource utilization.

The following is an example of output of the **show l2nat instance** and the **show l2nat statistics** commands:

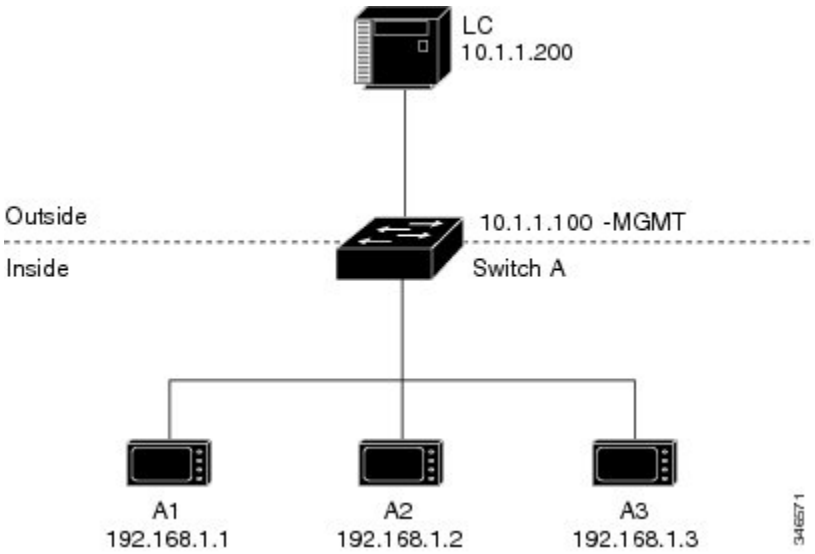
```
switch#show l2nat instance
l2nat instance test
fixup  : all
outside from host    10.10.10.200 to 192.168.1.200
inside  from host    192.168.1.1 to 10.10.10.1
l2nat instance test2
fixup  : all
inside  from host    1.1.1.1 to 2.2.2.2
outside from host    2.2.2.200 to 1.1.1.200

Switch#show l2nat interface
FOLLOWING INSTANCE(S) AND VLAN(s) ATTACHED TO ALL INTERFACES
============================================================
l2nat Gi1/1 test
============================================================

Switch#show l2nat statistics

STATS FOR INSTANCE: test (IN PACKETS)

TRANSLATED STATS (IN PACKETS)
```

```
================================================================================
INTERFACE DIRECTION VLAN   TRANSLATED
Gi1/1     EGRESS    50    0
Gi1/1     INGRESS   50    0
--------------------------------------------------------------------------------

PROTOCOL FIXUP STATS (IN PACKETS)
================================================================================
INTERFACE DIRECTION VLAN    ARP
Gi1/1     REPLY    50    0
Gi1/1     REQUEST   50    0
--------------------------------------------------------------------------------

PER TRANSLATION STATS (IN PACKETS)
================================================================================
TYPE    DIRECTION SA/DA ORIGINAL IP     TRANSLATED IP   COUNT
OUTSIDE INGRESS   SA    10.10.10.200    192.168.1.200   0
OUTSIDE EGRESS    DA    192.168.1.200   10.10.10.200    0
INSIDE  EGRESS    SA    192.168.1.1     10.10.10.1      0
INSIDE  INGRESS   DA    10.10.10.1      192.168.1.1     0
--------------------------------------------------------------------------------
================================================================================

TOTAL TRANSLATIONS ENTRIES IN HARDWARE: 4
TOTAL INSTANCES ATTACHED : 1
================================================================================
GLOBAL NAT STATISTICS
================================================================================
Total Number of TRANSLATED NAT  Packets   = 0
Total Number of ARP      FIX UP Packets   = 0
================================================================================
ad
```

Basic Inside-to-Outside Communications: Example

In this example, A1 must communicate with a logic controller (LC) that is directly connected to the uplink port. A Layer 2 NAT instance is configured to provide an address for A1 on the outside network (10.1.1.1) and an address for the LC on the inside network (192.168.1.250).

Figure 134: Basic Inside-to-Outside Communications



Now this communication can occur:

1. A1 sends an ARP request: SA: 192.168.1.1DA: 192.168.1.250.
2. Cisco Switch A fixes up the ARP request:SA:10.1.1.1DA: 10.1.1.200.
3. LC receives the request and learns the MAC Address of 10.1.1.1.
4. LC sends a response:SA: 10.1.1.200DA: 10.1.1.1.
5. Cisco Switch A fixes up the ARP response:SA: 192.168.1.250DA: 192.168.1.1.
6. A1 learns the MAC address for 192.168.1.250, and communication starts.

Note

- The management interface of the switch must be on a different VLAN from the inside network 192.168.1.x.
- See the section Basic Inside-to-Outside Communications: Configuration, on page 1819 for the tasks to configure the example in this section.

Basic Inside-to-Outside Communications: Configuration

This section contains the steps to configure inside-to-outside communications as described in the preceding section. You create the Layer 2 NAT instance, add two translation entries, and then apply the instance to the interface. ARP fixups are enabled by default.

Before you begin

Read and understand the content in the section Basic Inside-to-Outside Communications: Example, on page 1818.

Procedure

Step 1 Enter configuration mode.

Example:

```
switch# configure
```

Step 2 Create a new Layer 2 NAT instance called A-LC.

Example:

```
switch(config)# l2nat instance A-LC
```

Step 3 Translate A1's inside address to an outside address.

Example:

```
switch(config-l2nat)# inside from host 192.168.1.1 to 10.1.1.1
```

Step 4 Translate A2's inside address to an outside address.

Example:

```
switch(config-l2nat)# inside from host 192.168.1.2 to 10.1.1.2
```

Step 5 Translate A3's inside address to an outside address.

Example:

```
switch(config-l2nat)# inside from host 192.168.1.3 to 10.1.1.3
```

Step 6 Translate the LC outside address to an inside address.

Example:

```
switch(config-l2nat)# outside from host 10.1.1.200 to 192.168.1.250
```

Step 7 Exit config-l2nat mode.

Example:

```
switch(config-l2nat)# exit
```

Step 8 Access interface configuration mode for the uplink port.

Example:

```
witch(config)# interface Gi1/1
```

Step 9 Apply this Layer 2 NAT instance to the native VLAN on this interface.

Example:

```
switch(config-if)# l2nat A-LC
```

Note
For tagged traffic on a trunk, add the VLAN number when attaching the instance to an interface as follows:

l2nat *instance vlan*

Step 10 Return to privileged EXEC mode.

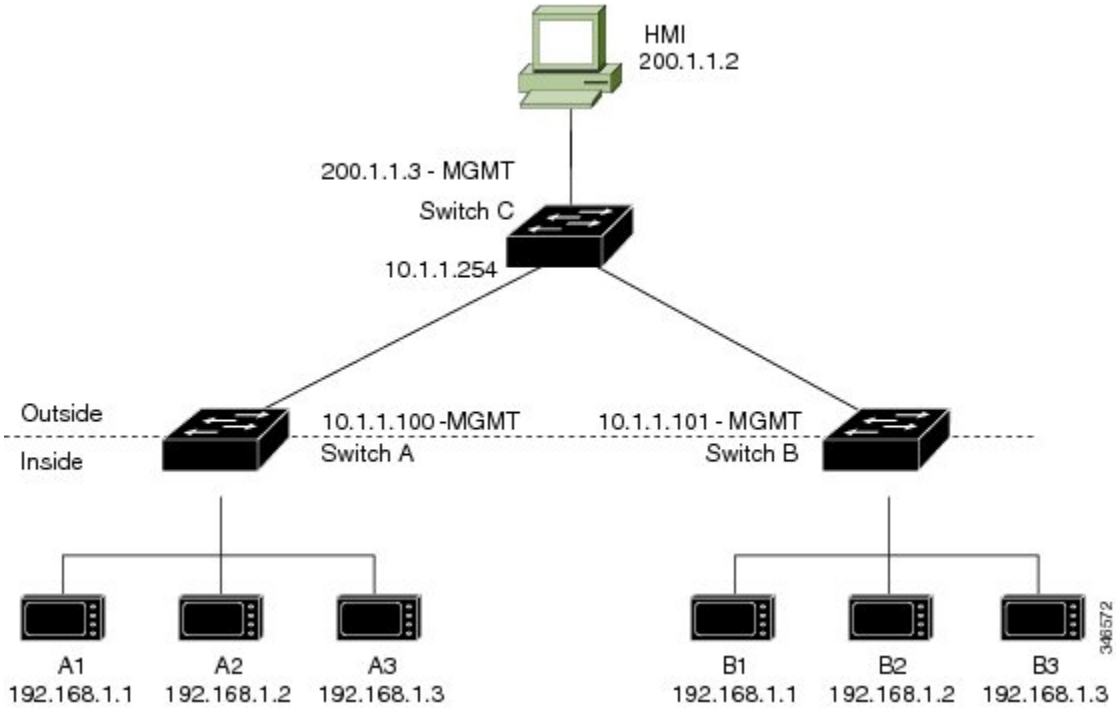
Example:

```
switch# end
```

Duplicate IP Addresses Example

In this scenario, two machine nodes are preconfigured with addresses in the 192.168.1.x space. Layer 2 NAT translates these addresses to unique addresses on separate subnets of the outside network. In addition, for machine-to-machine communications, the Node A machines need unique addresses on the Node B space and the Node B machines need unique addresses in the Node A space.

Figure 135: Duplicate IP Addresses



- Switch C needs an address in the 192.168.1.x space. When packets come into Node A or Node B, the 10.1.1.254 address of Switch C is translated to 192.168.1.254. When packets leave Node A or Node B, the 192.168.1.254 address of Switch C is translated to 10.1.1.254.
- Node A and Node B machines need unique addresses in the 10.1.1.x space. For quick configuration and ease of use, the 10.1.1.x space is divided into subnets: 10.1.1.0, 10.1.1.16, 10.1.1.32, and so on. Each subnet can then be used for a different node. In this example, 10.1.1.16 is used for Node A, and 10.1.1.32 is used for Node B.
- Node A and Node B machines need unique addresses to exchange data. The available addresses are divided into subnets. For convenience, the 10.1.1.16 subnet addresses for the Node A machines are translated to 192.168.1.16 subnet addresses on Node B. The10.1.1.32 subnet addresses for the Node B machines are translated to 192.168.1.32 addresses on Node A.
- Machines have unique addresses on each network:

Table 127: Translated IP Addresses

Node	Address in Node A	Address in Outside Network	Address in Node B
Switch A network address	192.168.1.0	10.1.1.16	192.168.1.16
A1	192.168.1.1	10.1.1.17	192.168.1.17
A2	192.168.1.2	10.1.1.18	192.168.1.18
A3	192.168.1.3	10.1.1.19	192.168.1.19
Cisco Switch B network address	192.168.1.32	10.1.1.32	192.168.1.0
B1	192.168.1.33	10.1.1.33	192.168.1.1
B2	192.168.1.34	10.1.1.34	192.168.1.2
B3	192.168.1.35	10.1.1.35	192.168.1.3
Switch C	192.168.1.254	10.1.1.254	192.168.1.254

Duplicate IP Addresses Configuration: Switch A

This section provides the steps for configuring Layer 2 NAT to translate the duplicated IP address of one machine node in an inside network to a unique address on a subnet of an outside network. This procedure is for Switch A in the section Duplicate IP Addresses Example, on page 1821.

Before you begin

Read and understand the content in the section Duplicate IP Addresses Example, on page 1821.

Procedure

Step 1 Enter global configuration mode.

Example:

```
switch# configure
```

Step 2 Create a new Layer 2 NAT instance called A-Subnet.

Example:

```
switch(config)# l2nat instance A-Subnet
```

Step 3 Translate the Node A machines' inside addresses to addresses in the 10.1.1.16 255.255.255.240 subnet.

Example:

```
switch(config-l2nat)# inside from network 192.168.1.0 to 10.1.1.16 mask
255.255.255.240
```

Step 4 Translate the outside address of Switch C to an inside address.

Example:

```
switch(config-l2nat)# outside from host 10.1.1.254 to 192.168.1.254
```

Step 5 Translate the Node B machines' outside addresses to their inside addresses.

Example:

```
switch(config-l2nat)# outside from host 10.1.1.32 to 192.168.1.32
outside from host 10.1.1.33 to 192.168.1.33
outside from host 10.1.1.34 to 192.168.1.34
outside from host 10.1.1.35 to 192.168.1.35
```

Step 6 Exits config-l2nat mode.

Example:

```
switch(config-l2nat)# exit
```

Step 7 Access interface configuration mode for the uplink port.

Example:

```
switch(config)# interface Gi1/1
```

Step 8 Apply this Layer 2 NAT instance to the native VLAN on this interface.

Example:

```
switch(config-if)# l2nat A-Subnet
```

Note
For tagged traffic on a trunk, add the VLAN number when attaching the instance to an interface as follows:

l2nat *instance vlan*

Step 9 Return to privileged EXEC mode.

Example:

```
switch# end
```

What to do next

Configure Layer 2 NAT to translate the duplicated IP address of Switch B in the section Duplicate IP Addresses Example, on page 1821. See Duplicate IP Addresses Configuration: Switch B, on page 1823.

Duplicate IP Addresses Configuration: Switch B

This section provides the steps for configuring Layer 2 NAT to translate the duplicated IP address of one machine node in an inside network to a unique address on a subnet of an outside network. This procedure is for Switch B in the section Duplicate IP Addresses Example, on page 1821.

Before you begin

Read and understand the content in the section Duplicate IP Addresses Example, on page 1821.

Procedure

Step 1 Enter global configuration mode.

Example:

```
switch# configure
```

Step 2 Create a new Layer 2 NAT instance called B-Subnet.

Example:

```
switch(config)# l2nat instance B-Subnet
```

Step 3 Translate the Node B machines' inside addresses to addresses in the 10.1.1.32 255.255.255.240 subnet.

Example:

```
switch(config-l2nat)# inside from network 192.168.1.0 to 10.1.1.32
255.255.255.240
```

Step 4 Translate the outside address of Switch C to an inside address.

Example:

```
switch(config-l2nat)# outside from host 10.1.1.254 to
```

Step 5 Translate the Node A machines' outside addresses to their inside addresses.

Example:

```
switch(config-l2nat)# outside from host 10.1.1.16 to 192.168.1.16
outside from host 10.1.1.17 to 192.168.1.17
outside from host 10.1.1.18 to 192.168.1.18
outside from host 10.1.1.19 to 192.168.1.19
```

Step 6 Exit config-l2nat mode.

Example:

```
switch(config-l2nat)# exit
```

Step 7 Access interface configuration mode for the uplink port.

Example:

```
switch(config)# interface Gi1/1
```

Step 8 Apply this Layer 2 NAT instance to the native VLAN on this interface.

Example:

```
switch(config-if)# l2nat name1
```

Note
For tagged traffic on a trunk, add the VLAN number when attaching the instance to an interface as follows:

l2nat *instance vlan*

Step 9 Show the configuration details for the specified Layer 2 NAT instance.

Example:

```
switch# show l2nat instance name1
```

Step 10 Show Layer 2 NAT statistics.

Example:

```
switch# show l2nat statistics
```

Step 11 Return to privileged EXEC mode.

Example:

```
switch# end
```

CHAPTER 130

Layer 3 Network Address Translation

Network Address Translation

Network Address Translation (NAT) is designed for IP address conservation. It enables private IP networks that use unregistered IP addresses to connect to the Internet. NAT operates on a device, usually connecting two networks together, and translates the private (not globally unique) addresses in the internal network into global routable addresses. It does so before packets are forwarded onto another network.

NAT can be configured to advertise only one address for the entire network to the outside world. This ability provides more security by effectively hiding the entire internal network behind that one address. NAT offers the dual functions of security and address conservation and is typically implemented in remote-access environments.

NAT is also used at the enterprise edge to allow internal users access to the Internet and to allow Internet access to internal devices such as mail servers.

Finding Feature Information

Your software release may not support all the features described in this document. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this chapter.

Use the Cisco Feature Navigator to find information about platform support and Cisco software image support. To access the Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Benefits of Configuring NAT

Configuring NAT provides the following benefits:

- NAT Resolves the problem of IP depletion.

  NAT allows organizations to resolve the problem of IP address depletion when they have existing networks and need to access the Internet. Sites that do not yet possess Network Information Center (NIC)-registered IP addresses must acquire IP addresses. In such cases, if more than 254 clients are present or are planned, the scarcity of Class B addresses becomes a serious issue. NAT addresses these issues by mapping thousands of hidden internal addresses to a range of easy-to-get Class C addresses.

- NAT provides a layer of security by preventing the client IP address from being exposed to the outside network.

  Sites that already have registered IP addresses for clients on an internal network may want to hide those addresses from the Internet so that hackers cannot directly attack clients. With client addresses hidden, a degree of security is established. NAT gives LAN administrators complete freedom to expand Class A addressing, which is drawn from the reserve pool of the Internet Assigned Numbers Authority. The expansion of Class A addresses occurs within the organization without a concern for addressing changes at the LAN or the Internet interface.

- Cisco software can selectively or dynamically perform NAT. This flexibility allows network administrator to use RFC 1918 addresses or registered addresses.

- NAT is designed for use on a variety of devices for IP address simplification and conservation. In addition, NAT allows the selection of internal hosts that are available for translation.

- A significant advantage of NAT is that it can be configured without requiring any changes to devices other than to those few devices on which NAT will be configured.

How NAT Works

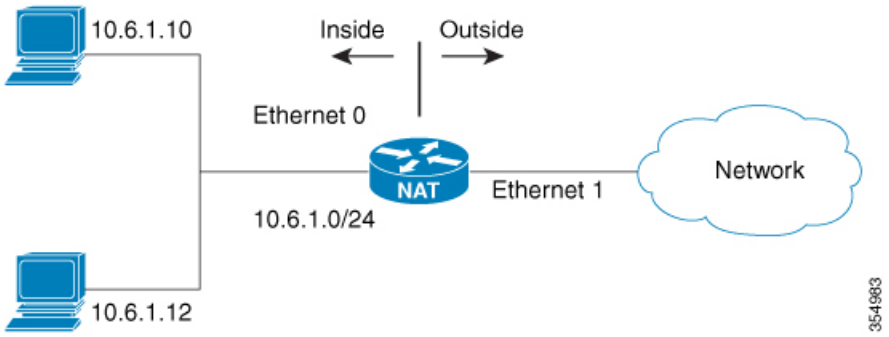
A device that is configured with NAT has at least one interface to the inside network and one to the outside network. In a typical environment, NAT is configured at the exit device between a stub domain and the backbone. When a packet leaves the domain, NAT translates the locally significant source address into a globally unique address. When a packet enters the domain, NAT translates the globally unique destination address into a local address.

Multiple inside networks could be connected to the device and similarly there might exist multiple exit points from the device towards outside networks. If NAT cannot allocate an address because it has run out of addresses,

it drops the packet and sends an Internet Control Message Protocol (ICMP) host unreachable packet to the destination.

Translation and forwarding are performed in the hardware switching plane, improving the overall throughput performance. For more details on performance, see the section Performance and Scale Numbers for NAT, on page 1836.

Figure 136: NAT



Uses of NAT

You can use NAT in the following scenarios:

- To connect to the Internet when only a few of your hosts have globally unique IP address.

  NAT is configured on a device at the border of a stub domain (referred to as the inside network) and a public network such as the Internet (referred to as the outside network). NAT translates internal local addresses to globally unique IP addresses before sending packets to the outside network.

  As a solution to the connectivity problem, NAT is practical only when relatively few hosts in a stub domain communicate outside of the domain at the same time. In such cases, only a small subset of the IP addresses in the domain must be translated into globally unique IP addresses when outside communication is necessary, and these addresses can be reused.

- To renumber:

  Instead of changing the internal addresses, which can be a considerable amount of work, you can translate them by using NAT.

NAT Inside and Outside Addresses

The term *inside* in a NAT context refers to networks owned by an organization that must be translated. When NAT is configured, hosts within this network have addresses in one space (known as the local address space) that appears to those outside the network as being in another space (known as the global address space).

Similarly, the term *outside* refers to those networks to which the stub network connects, and which are generally not under the control of an organization. Hosts in outside networks can also be subject to translation, and can thus have local and global addresses.

NAT uses the following definitions:

- Inside local address: an IP address that is assigned to a host on the inside network. The address is probably not a routable IP address assigned by NIC or service provider.
- Inside global address: a global routable IP address (assigned by the NIC or service provider) that represents one or more inside local IP addresses to the outside world.
- Outside local address: the IP address of an outside host as it appears to the inside network. Not necessarily a routable IP address, it is allocated from the address space that is routable on the inside.
- Outside global address: the IP address assigned to a host on the outside network by the owner of the host. The address is allocated from a globally routable address or network space.
- Inside Source Address Translation: translates an inside local address to inside global address.
- Outside Source Address Translation: translates the outside global address to outside local address.
- Static Port Translation: translates the IP address and port number of an inside/outside local address to the IP address and port number of the corresponding inside/outside global address.
- Static Translation of a given subnet: translates a specified range of subnets of an inside/outside local address to the corresponding inside/outside global address.
- Half Entry: represents a mapping between the local and global address/ports and is maintained in the translation database of NAT module. A half entry may be created statically or dynamically based on the configured NAT rule.
- Full Entry/Flow entry: represents a unique flow corresponding to a given session. In addition to the local to global mapping, it also maintains the destination information which fully qualifies the given flow. A Full entry is always created dynamically and maintained in the translation database of NAT module.

Types of NAT

You can configure NAT such that it advertises only a single address for your entire network to the outside world. The configuration effectively hides the internal network from the world, giving you some additional security.

The types of NAT include:

- Static address translation (static NAT): Allows one-to-one mapping between local and global addresses.
- Dynamic address translation (dynamic NAT): Maps unregistered IP addresses to registered IP addresses from a pool of registered IP addresses.
- Overloading / PAT: Maps multiple unregistered IP addresses to a single registered IP address (many to one) using different Layer 4 ports. This method is also known as Port Address Translation (PAT). By using overloading, thousands of users can be connected to the Internet by using only one real global IP address.

Using NAT to Route Packets to the Outside Network (Inside Source Address Translation)

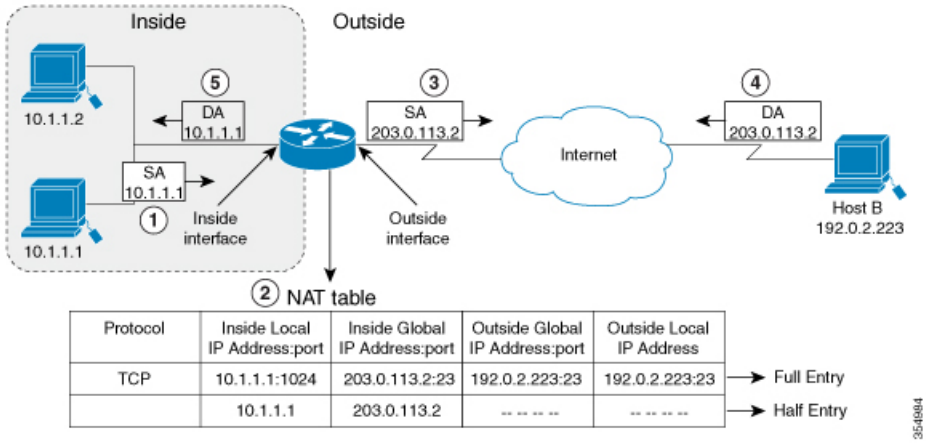
You can translate unregistered IP addresses into globally unique IP addresses when communicating outside your network.

You can configure static or dynamic inside source address translation as follows:

- Static translation establishes a one-to-one mapping between the inside local address and an inside global address. Static translation is useful when a host on the inside must be accessible by a fixed address from the outside. You can enable Static translation by configuring a static NAT rule as explained in the x section.
- Dynamic translation establishes a mapping between an inside local address and a pool of global addresses dynamically. Dynamic translation can be enabled by configuring a dynamic NAT rule and the mapping is established based on the result of the evaluation of the configured rule at run-time. You can employ an Access Control List (ACL), both Standard and Extended ACLs, to specify the inside local address. The inside global address can be specified through an address pool or an interface. Dynamic translation is enabled by configuring a dynamic rule as explained in the section Configuring Dynamic Translation of Inside Source Addresses, on page 1838.

The following figure illustrates a device that is translating a source address inside a network to a source address outside the network.

Figure 137: NAT Inside Source Translation



The following process describes the inside source address translation, as shown in the preceding figure:

1. The user at host 10.1.1.1 opens a connection to Host B in the outside network.
2. NAT module intercepts the corresponding packet and attempts to translate the packet.

   The following scenarios are possible based on the presence or absence of a matching NAT rule:

   - If a matching static translation rule exists, the packet gets translated to the corresponding inside global address. Otherwise, the packet is matched against the dynamic translation rule, and in the event of a successful match, it gets translated to the corresponding inside global address. The NAT

module inserts a fully qualified flow entry corresponding to the translated packet, into its translation database. This facilitates fast translation and forwarding of the packets corresponding to this flow, in either direction.

- The packet gets forwarded without any address translation in the absence of a successful rule match.
- The packet is dropped in the event of failure to obtain a valid inside global address even-though we have a successful rule match.

Note If an ACL is employed for dynamic translation, NAT evaluates the ACL and ensures that only the packets that are permitted by the given ACL are considered for translation.

3. The device replaces the inside local source address of host 10.1.1.1 with the inside global address of the translation, 203.0.113.2, and forwards the packet.
4. Host B receives the packet and responds to host 10.1.1.1 by using the inside global IP destination address (DA) 203.0.113.2.
5. The response packet from host B would be destined to the inside global address. The NAT module intercepts this packet and translates it back to the corresponding inside local address with the help of the flow entry that has been set up in the translation database.

Host 10.1.1.1 receives the packet and continues the conversation. The device performs Step 2 to Step 5 for each packet that it receives.

Outside Source Address Translation

You can translate the source address of the IP packets that travel from outside of the network to inside the network. This type of translation is usually employed in conjunction with inside source address translation to interconnect overlapping networks.

This process is explained in the section Configuring Translation of Overlapping Networks, on page 1843.

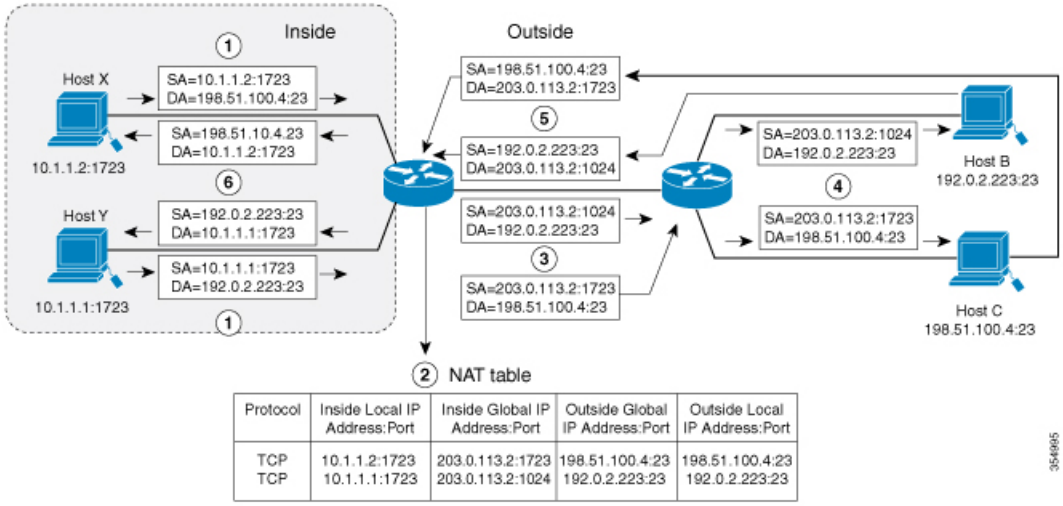
Port Address Translation

You can conserve addresses in the inside global address pool by allowing a device to use one global address for many local addresses. This type of NAT configuration is called overloading or port address translation (PAT).

When overloading is configured, the device maintains enough information from higher-level protocols (for example, TCP or UDP port numbers) to translate the global address back to the correct local address. When multiple local addresses map to one global address, the TCP or UDP port numbers of each inside host distinguish between the local addresses.

The following figure illustrates a NAT operation when an inside global address represents multiple inside local addresses. The TCP port numbers act as differentiators.

Figure 138: PAT / NAT Overloading Inside Global Addresses



The device performs the following process in the overloading of inside global addresses, as shown in the figure above. Both Host B and Host C believe that they are communicating with a single host at address 203.0.113.2. However, they are actually communicating with different hosts; the port number is the differentiator. In fact, many inside hosts can share the inside global IP address by using many port numbers.

1. The user at host 10.1.1.1:1723 opens a connection to Host B and the user at host 10.1.1.2:1723 opens a connection to Host C.
2. NAT module intercepts the corresponding packets and attempts to translate the packets.

   Based on the presence or absence of a matching NAT rule the following scenarios are possible:

   - If a matching static translation rule exists, then it takes precedence and the packets are translated to the corresponding global address. Otherwise, the packets are matched against dynamic translation rule and in the event of a successful match, they are translated to the corresponding global address. NAT module inserts a fully qualified flow entry corresponding to the translated packets, into its translation database, to facilitate fast translation and forwarding of the packets corresponding to this flow, in either direction.
   - The packets are forwarded without any address translation in the absence of a successful rule match.
   - The packets are dropped in the event of failure to obtain a valid inside global address even though we have a successful rule match.
   - Because this is a PAT configuration, transport ports help translate multiple flows to a single global address. (In addition to source address, the source port is also subjected to translation and the associated flow entry maintains the corresponding translation mappings.)

3. The device replaces inside local source address/port 10.1.1.1/1723 and 10.1.1.2/1723 with the corresponding selected global address/port 203.0.113.2/1024 and 203.0.113.2/1723 respectively and forwards the packets.
4. Host B receives the packet and responds to host 10.1.1.1 by using the inside global IP address 203.0.113.2, on port 1024. Host C receives the packet and responds to host 10.1.1.2 using the inside global IP address 203.0.113.2, on port 1723.

5. When the device receives the packets with the inside global IP address, it performs a NAT table lookup; the inside global address and port, and the outside address and port as keys; translates the addresses to the inside local addresses 10.1.1.1:1723 / 10.1.1.2:1723 and forwards the packets to host 10.1.1.1. and 10.1.1.2 respectively.

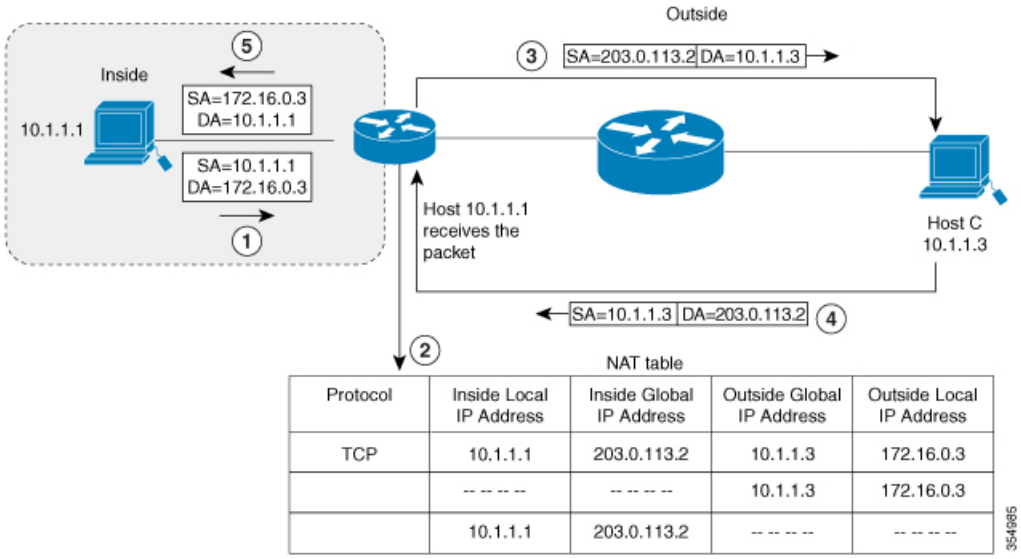
Host 10.1.1.1 and Host 10.1.1.2 receive the packet and continue the conversation. The device performs Step 2 to Step 5 for each packet it receives.

Overlapping Networks

Use NAT to translate IP addresses if the IP addresses that you use are not legal or officially assigned. Overlapping networks result when you assign an IP address to a device on your network that is already legally owned and assigned to a different device on the Internet or outside the network.

The following figure depicts overlapping networks: the inside network and outside network both have the same local IP addresses (10.1.1.x). You need network connectivity between such overlapping address spaces with one NAT device to translate the address of a remote peer (10.1.1.3) to a different address from the perspective of the inside.

Figure 139: NAT Translating Overlapping Addresses



Notice that the inside local address (10.1.1.1) and the outside global address (10.1.1.3) are in the same subnet. To translate the overlapping address, first, the inside source address translation happens with the inside local address getting translated to 203.0.113.2 and a half entry is created in the NAT table. On the Receiving side, the outside source address is translated to 172.16.0.3 and another half entry is created. The NAT table is then updated with a full entry of the complete translation.

The following steps describe how a device translates overlapping addresses:

1. Host 10.1.1.1 opens a connection to 172.16.0.3.
2. The NAT module sets up the translation mapping of the inside local and global addresses to each other and the outside global and local addresses to each other.

3. The Source Address (SA) is replaced with inside global address and the Destination Address (DA) is replaced with outside global address.
4. Host C receives the packet and continues the conversation.
5. The device does a NAT table lookup, replaces the DA with inside local address, and replaces the SA with outside local address.
6. Host 10.1.1.1 receives the packet and the conversation continues using this translation process.

Limitations of NAT

- Some NAT operations are currently not supported in the hardware data plane. The following are such operations that are carried out in the relatively slower software data plane:
  - Translation of Internet Control Message Protocol (ICMP) packets
  - Translation of packets that require application layer gateway (ALG) processing
  - Packets that require both inside and outside translation
- The maximum number of sessions that can be translated and forwarded in the hardware in an ideal setting is limited to 192. Additional flows that require translation are handled in the software data plane at a reduced throughput.

Note Each translation consumes two entries in TCAM.

- A configured NAT rule might fail to get programmed into the hardware owing to resource constraint. This could result in packets that correspond to the given rule to get forwarded without translation.
- ALG support is currently limited to FTP, TFTP, and ICMP protocols. Also, although TCP SYN, TCP FIN and TCP RST are not part of ALG traffic, they are processed as part of ALG traffic.
- Dynamically created NAT flows age out after a period of inactivity. The number of NAT flows whose activity can be tracked is limited to 192.
- Port channel is not supported in NAT configuration.
- NAT does not support translation of fragmented packets.
- Explicit deny access control entry (ACE) in NAT ACL is not supported. Only explicit permit ACE is supported.
- NAT and PBR share the same TCAM space and they cannot co-exist.
- NAT configuration must be done without using route maps because route mapped NAT is not supported.
- NAT is not supported for multicast packets.

Performance and Scale Numbers for NAT

The maximum number of bidirectional NAT flows supported in hardware is limited to 192.

Address Only Translation

Note Using Address Only Translation optimizes the handling of flows and enhances the scale of the NAT feature.

You can use Address only Translation (AOT) functionality in situations that require only the address fields to be translated and not the transport ports. In such settings, enabling AOT functionality significantly increases the number of flows that can be translated and forwarded in the hardware at line-rate. This improvement is brought about by optimizing the usage of various hardware resources associated with translation and forwarding.

A typical NAT focused resource allocation scheme sets aside 384 TCAM entries for performing hardware translation. This places a strict upper limit on the number of flows that can be translated and forwarded at line-rate. Under AOT scheme, the usage of TCAM resource is highly optimized thereby enabling the accommodation of more number of flows in the TCAM tables and this provides a significant improvement in the hardware translation and forwarding scale.

AOT can be very effective in situations where majority of the flows are destined to a single or a small set of destinations. Under such favorable conditions, AOT can potentially enable line-rate translation and forwarding of all the flows originating from one or more given end-points. AOT functionality is disabled by default. It can be enabled using the **no ip nat create flow-entries** command. The existing dynamic flow can be cleared using the **clear ip nat translation** command. The AOT feature can be disabled using the **ip nat create flow-entries** command.

Restrictions for Address Only Translation

- AOT feature is expected to function correctly only in translation scenarios corresponding to simple inside static and inside dynamic rules. The simple static rule must be of the type **ip nat inside source static** *local-ip global-ip*, and the dynamic rule must be of the type **ip nat inside source list** *access-list* **pool** *name*.
- When AOT is enabled, the **show ip nat translation** command will not give visibility into all the NAT flows being translated and forwarded.

Configuring NAT

The tasks described in this section will help you configure NAT. Based on the desired configuration, you may need to configure more than one task.

Configuring Static Translation of Inside Source Addresses

Configure static translation of inside source address to allow one-to-one mapping between an inside local address and an inside global address. Static translation is useful when a host on the inside must be accessible by a fixed address from the outside.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	Use any of the following three commands depending on the requirement: • **ip nat inside source static** *local-ip global-ip* `Switch(config)# ip nat inside source static 10.10.10.1 172.16.131.` • **ip nat inside source static** *protocol local-ip port global-ip port* `Switch(config)# ip nat inside source static tcp 10.10.10.1 1234 172.16.131.1 5467` • **ip nat inside source static network** *local-ip global-ip* {**prefix_len** *len* | **subnet** *subnet-mask*} `Switch(config)# ip nat inside source static network 10.10.10.1 172.16.131.1 prefix_len 24`	Establishes static translation between an inside local address and an inside global address. Establishes a static port translation between an inside local address and an inside global address. Establishes a static translation between an inside local address and an inside global address. You can specify a range of subnets to be translated to the inside global address, wherein the host portion of the IP address gets translated and the network portion of the IP remains the same.
Step 4	**interface** *type number* **Example:** `Switch(config)# interface GigabitEthernet 1/1`	Specifies an interface and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for an interface.

	Command or Action	Purpose
Step 6	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 7	**exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	**interface** *type number* **Example:** `Switch(config)# interface GigabitEthernet 1/2`	Specifies a different interface and enters interface configuration mode.
Step 9	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)# ip address 172.31.232.182 255.255.255.240`	Sets a primary IP address for an interface.
Step 10	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 11	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Dynamic Translation of Inside Source Addresses

Dynamic translation establishes a mapping between an inside local address and a pool of global addresses dynamically. Dynamic translation can be enabled by configuring a dynamic NAT rule and the mapping is established based on the result of the evaluation of the configured rule at run-time. You can employ an ACL to specify the inside local address and the inside global address can be specified through an address pool or an interface.

Dynamic translation is useful when multiple users on a private network need to access the Internet. The dynamically configured pool IP address may be used as needed and is released for use by other users when access to the internet is no longer required.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* **netmask** *netmask* | **prefix-length** *prefix-length* **Example:** `Switch(config)# ip nat pool net-208 172.16.233.208 172.16.233.223 prefix-length 28`	Defines a pool of global addresses to be allocated as needed.
Step 4	**access-list** *access-list-number* **permit** *source [source-wildcard]* **Example:** `Switch(config)# access-list 1 permit 192.168.34.0 0.0.0.255`	Defines a standard access list permitting those addresses that are to be translated.
Step 5	**ip nat inside source list** *access-list-number* **pool** *name* **Example:** `Switch(config)# ip nat inside source list 1 pool net-208`	Establishes dynamic source translation, specifying the access list defined in Step 4.
Step 6	**interface** *type number* **Example:** `Switch(config)# interface GigabitEthernet 1/1`	Specifies an interface and enters interface configuration mode.
Step 7	**ip address** *ip-address mask* **Example:** `Switch(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for the interface.
Step 8	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 9	**exit** **Example:** `Switch(config-if)#exit`	Exits the interface configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Switch(config)# interface GigabitEthernet 1/2`	Specifies an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 11	**ip address** *ip-address mask* **Example:** `Switch(config-if)# ip address 172.16.232.182 255.255.255.240`	Sets a primary IP address for the interface.
Step 12	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 13	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring PAT

Perform this task to allow your internal users access to the Internet and conserve addresses in the inside global address pool using overloading of global addresses.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* **netmask** *netmask* | **prefix-length** *prefix-length* **Example:** `Switch(config)# ip nat pool net-208 192.168.202.129 192.168.202.158 netmask 255.255.255.224`	Defines a pool of global addresses to be allocated as needed.
Step 4	**access-list** *access-list-number* **permit** *source [source-wildcard]* **Example:** `Switch(config)# access-list 1 permit 192.168.201.30 0.0.0.255`	Defines a standard access list permitting those addresses that are to be translated. The access list must permit only those addresses that are to be translated. (Remember that there is an implicit “deny all” at the end of each access list.) Use of an access list that is too permissive can lead to unpredictable results.

	Command or Action	Purpose
Step 5	**ip nat inside source list** *access-list-number* **pool** *name* **overload** **Example:** `Switch(config)# ip nat inside source list 1 pool net-208 overload`	Establishes dynamic source translation with overloading, specifying the access list defined in Step 4.
Step 6	**interface** *type number* **Example:** `Switch(config)# interface GigabitEthernet 1/1`	Specifies an interface and enters interface configuration mode.
Step 7	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)# ip address 192.168.201.1 255.255.255.240`	Sets a primary IP address for an interface.
Step 8	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 9	**exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Switch(config)# interface GigabitEthernet 1/2`	Specifies a different interface and enters interface configuration mode.
Step 11	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)#  ip address 192.168.201.29 255.255.255.240`	Sets a primary IP address for an interface.
Step 12	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 13	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring NAT of External IP Addresses Only

By default, NAT translates the addresses embedded in the packet pay-load as explained in the section Using Application-Level Gateways with NAT, on page 1846. There might be situations where the translation of the embedded address is not desirable and in such cases, NAT can be configured to translate the external IP address only.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static network** *local-ip global-ip* [**no-payload**]} **Example:** `Device(config)# ip nat inside source static network 10.1.1.1 192.168.251.0/24 no-payload`	Disables the network packet translation on the inside host device.
Step 4	**ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static** {**tcp** | **udp**} *local-ip local-port global-ip global-port* [**no-payload**]} **Example:** `Device(config)# ip nat inside source static tcp 10.1.1.1 2000 192.168.1.1 2000 no-payload`	Disables port packet translation on the inside host device.
Step 5	**ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static** [**network**] *local-network-mask global-network-mask* [**no-payload**]} **Example:** `Device(config)# ip nat inside source static 10.1.1.1 192.168.1.1 no-payload`	Disables packet translation on the inside host device.
Step 6	**ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** *local-ip global-ip* [**no-payload**]}	Disables packet translation on the outside host device.

	Command or Action	Purpose
	Example: `Device(config)# ip nat outside source static 10.1.1.1 192.168.1.1 no-payload`	
Step 7	**ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** {**tcp** | **udp**} *local-ip local-port global-ip global-port* [**no-payload**]} **Example:** `Device(config)# ip nat outside source static tcp 10.1.1.1 20000 192.168.1.1 20000 no-payload`	Disables port packet translation on the outside host device.
Step 8	**ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** [**network**] *local-network-mask global-network-mask* [**no-payload**]} **Example:** `Device(config)# ip nat outside source static network 10.1.1.1 192.168.251.0/24 no-payload`	Disables network packet translation on the outside host device.
Step 9	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 10	**show ip nat translations** [**verbose**] **Example:** `Device# show ip nat translations`	Displays active NAT.

Configuring Translation of Overlapping Networks

Configure static translation of overlapping networks if your IP addresses in the stub network are legitimate IP addresses belonging to another network and you want to communicate with those hosts or routers using static translation.

Note For a successful NAT outside translation, the device should be configured with a route for the outside local address. You can configure the route either manually or using the **add-route** option associated with **ip nat outside source** {**static** | **list**} command. We recommend that you use the **add-route** option to enable automatic creation of the route.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	
Step 3	**ip nat inside source static** *local-ip global-ip* **Example:** `Switch(config)# ip nat inside source static 10.1.1.1 203.0.113.2`	Establishes static translation between an inside local address and an inside global address.
Step 4	**ip nat outside source static** *local-ip global-ip* **Example:** `Switch(config)# ip nat outside source static 172.16.0.3 10.1.1.3`	Establishes static translation between an outside local address and an outside global address.
Step 5	**interface** *type number* **Example:** `Switch(config)# interface GigabitEthernet 1/1`	Specifies an interface and enters interface configuration mode.
Step 6	**ip address** *ip-address mask* **Example:** `Switch(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for an interface.
Step 7	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Marks the interface as connected to the inside.
Step 8	**exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**interface** *type number* **Example:** `Switch(config)# interface GigabitEthernet 1/2`	Specifies a different interface and enters interface configuration mode.
Step 10	**ip address** *ip-address mask* **Example:**	Sets a primary IP address for an interface.

	Command or Action	Purpose
	`Switch(config-if)#  ip address 172.16.232.182 255.255.255.240`	
Step 11	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Marks the interface as connected to the outside.
Step 12	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Address Translation Timeouts

You can configure address translation timeouts based on your NAT configuration.

By default, dynamically created translation entries time-out after a period of inactivity to enable the efficient use of various resources. You can change the default values on timeouts, if necessary. The following are the default time-out configurations associated with major translation types:

- Established TCP sessions: 24 hours
- UDP flow: 5 minutes
- ICMP flow: 1 minute

The default timeout values are adequate to address the timeout requirements in most of the deployment scenarios. However, these values can be adjusted/fine-tuned as appropriate. It is recommended not to configure very small timeout values (less than 60 seconds) as it could result in high CPU usage. Refer the x section for more information.

Based on your configuration, you can change the timeouts described in this section.

- If you need to quickly free your global IP address for a dynamic configuration, configure a shorter timeout than the default timeout, by using the **ip nat translation timeout** command. However, the configured timeout should be longer than the other timeouts configured using commands specified in the following steps.
- If a TCP session is not properly closed by a finish (FIN) packet from both sides or during a reset, change the default TCP timeout by using the **ip nat translation tcp-timeout** command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Switch# configure terminal`	
Step 3	**ip nat translation** *seconds* **Example:** `Switch(config)# ip nat translation 300`	(Optional) Changes the amount of time after which NAT translations time out. The default timeout is 24 hours, and it applies to the aging time for half-entries.
Step 4	**ip nat translation udp-timeout** *seconds* **Example:** `Switch(config)# ip nat translation udp-timeout 300`	(Optional) Changes the UDP timeout value.
Step 5	**ip nat translation tcp-timeout** *seconds* **Example:** `Switch(config)# ip nat translation tcp-timeout 2500`	(Optional) Changes the TCP timeout value. The default is 24 hours.
Step 6	**ip nat translation finrst-timeout** *seconds* **Example:** `Switch(config)# ip nat translation finrst-timeout 45`	(Optional) Changes the finish and reset timeout value. finrst-timeout—The aging time after a TCP session receives both finish-in (FIN-IN) and finish-out (FIN-OUT) requests or after the reset of a TCP session.
Step 7	**ip nat translation icmp-timeout** *seconds* **Example:** `Switch(config)# ip nat translation icmp-timeout 45`	(Optional) Changes the ICMP timeout value.
Step 8	**ip nat translation syn-timeout** *seconds* **Example:** `Switch(config)# ip nat translation syn-timeout 45`	(Optional) Changes the synchronous (SYN) timeout value. The synchronous timeout or the aging time is used only when a SYN request is received on a TCP session. When a synchronous acknowledgment (SYNACK) request is received, the timeout changes to TCP timeout.
Step 9	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Using Application-Level Gateways with NAT

NAT performs translation services on any TCP/UDP traffic that does not carry source and destination IP addresses in the application data stream. Protocols that do not carry the source and destination IP addresses include the following:

- HTTP
- TFTP
- Telnet
- Archie
- Finger
- Network Time Protocol (NTP)
- Network File System (NFS)
- Remote login (rlogin)
- Remote shell (rsh)
- Remote copy (rcp)

NAT Application-Level Gateway (ALG) enables certain applications that carry address/port information in their payloads to function correctly across NAT domains. In addition to the usual translation of address/ports in the packet headers, ALGs take care of translating the address/ports present in the payload and setting up temporary mappings.

Best Practices for NAT Configuration

- In cases where both static and dynamic rules are configured, ensure that the local addresses specified in the rules do not overlap. If such an overlap is possible, then the ACL associated with the dynamic rule should exclude the corresponding addresses used by the static rule. Similarly, there must not be any overlap between the global addresses as this could lead to undesired behavior.
- Do not employ loose filtering such as **permit ip any any** in an ACL associated with NAT rule as this could result in unwanted packets being translated.
- Do not share an address pool across multiple NAT rules.
- Do not define the same inside global address in Static NAT and Dynamic Pool. This action can lead to undesirable results.
- Exercise caution while modifying the default timeout values associated with NAT. Small timeout values could result in high CPU usage.
- Exercise caution while manually clearing the translation entries as this could result in the disruption of application sessions.
- ALG packets traversing a NAT enabled interface will get punted to CPU, regardless of the packets being translated or not. Therefore, it is recommended to use dedicated interface(s) just for NAT traffic. For all other types of traffic that does not require NAT translation, use a different interface(s).

Troubleshooting NAT

This section explains the basic steps to troubleshoot and verify NAT.

- Clearly define what NAT is supposed to achieve.
- Verify that correct translation table exists using the **show ip nat translation** command.
- Verify that timer values are correctly configured using the **show ip nat translation verbose** command.
- Check the ACL values for NAT using the **show ip access-list** command.
- Check the overall NAT configuration using the **show ip nat statistics** command.
- Use the **clear ip nat translation** command to clear the NAT translational table entires before the timer expires.
- Use debug nat ip and debug nat ip detailed commands to debug NAT configuration.

For further information on Troubleshooting NAT, see Verifying NAT Operation and Basic NAT Troubleshooting on Cisco.com..

PART V

QoS

CHAPTER 131

Configuring Auto-QoS

Prerequisites for Auto-QoS

The prerequisites for auto-QoS are the same as the prerequisites for standard QoS.

Restrictions for Auto-QoS

The following are restrictions for auto-QoS:

- Auto-qos is not supported on SVI interfaces.
- Do not configure the **auto qos voip cisco-phone** option for IP phones that support video. This option causes DSCP markings of video packets to get overwritten, because these packets do not have Expedited Forwarding priority, which results in these packets getting classified in the class-default class.

Information About Configuring Auto-QoS

Auto-QoS Overview

You can use the auto-QoS feature to simplify the deployment of QoS features. Auto-QoS determines the network design and enables QoS configurations so that the switch can prioritize different traffic flows.

The switch employs the MQC model. This means that instead of using certain global configurations, auto-QoS applied to any interface on a switch configures several global class maps and policy maps.

Auto-QoS matches traffic and assigns each matched packet to qos-groups. This allows the output policy map to put specific qos-groups into specific queues, including into the priority queue.

QoS is needed in both directions, both on inbound and outbound. When inbound, the switch port needs to trust the DSCP in the packet (done by default). When outbound, the switch port needs to give voice packets "front of line" priority. If voice is delayed too long by waiting behind other packets in the outbound queue, the end host drops the packet because it arrives outside of the receive window for that packet.

Auto-QoS Compact Overview

When you enter an auto-QoS command, the switch displays all the generated commands as if the commands were entered from the CLI. You can use the auto-QoS compact feature to hide the auto-QoS generated commands from the running configuration. This would make it easier to comprehend the running-configuration and also help to increase efficient usage of memory.

Auto-QoS Global Configuration Templates

In general, an auto-QoS command generates a series of class maps that either match on ACLs or on DSCP and/or CoS values to differentiate traffic into application classes. An input policy is also generated, which matches the generated classes and in some cases, polices the classes to a set bandwidth. Eight egress-queue class maps are generated. The actual egress output policy assigns a queue to each one of these eight egress-queue class maps.

The auto-QoS commands only generate templates as needed. For example, the first time any new auto-QoS command is used, global configurations that define the eight queue egress service-policy are generated. From this point on, auto-QoS commands applied to other interfaces do not generate templates for egress queuing because all auto-QoS commands rely on the same eight queue models, which have already been generated from the first time a new auto-QoS command was used.

Auto-QoS Policy and Class Maps

After entering the appropriate auto-QoS command, the following actions occur:

- Specific class maps are created.
- Specific policy maps (input and output) are created.
- Policy maps are attached to the specified interface.
- Trust level for the interface is configured.

Effects of Auto-QoS on Running Configuration

When auto-QoS is enabled, the **auto qos** interface configuration commands and the generated global configuration are added to the running configuration.

The switch applies the auto-QoS-generated commands as if the commands were entered from the CLI. An existing user configuration can cause the application of the generated commands to fail or to be overridden by the generated commands. These actions may occur without warning. If all the generated commands are successfully applied, any user-entered configuration that was not overridden remains in the running configuration. Any user-entered configuration that was overridden can be retrieved by reloading the switch

without saving the current configuration to memory. If the generated commands are not applied, the previous running configuration is restored.

Effects of Auto-Qos Compact on Running Configuration

If auto-QoS compact is enabled:

- Only the auto-QoS commands entered from the CLI are displayed in running-config.
- The generated global and interface configurations are hidden.
- When you save the configuration, only the auto-qos commands you have entered are saved (and not the hidden configuration).
- When you reload the switch, the system detects and re-executes the saved auto-QoS commands and the AutoQoS SRND4.0 compliant config-set is generated.

Note Do not make changes to the auto-QoS-generated commands when auto-QoS compact is enabled, because user-modifications are overridden when the switch reloads.

When auto-qos global compact is enabled:

- **show derived-config** command can be used to view hidden Auto-QoS global Compact derived commands.
- Auto-QoS global Compact commands will not be stored to memory. They will be regenerated every time the switch is reloaded.
- When compaction is enabled, auto-qos generated commands should not be modified.
- If the interface is configured with auto-QoS and if Auto-QoS global Compact needs to be disabled, auto-QoS should be disabled at interface level first.

How to configure Auto-QoS

Configuring Auto-QoS

For optimum QoS performance, configure auto-QoS on all the devices in your network.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**interface** *interface-id* **Example:** `Device(config)# gigabitethernet 1/1`	Specifies the port that is connected to a VoIP port, video device, or the uplink port that is connected to another trusted switch or router in the network interior, and enters the interface configuration mode.
Step 3	Depending on your auto-QoS configuration, use one of the following commands: • **auto qos voip** {**cisco-phone** | **cisco-softphone** | **trust**} • **auto qos video** {**cts** | **ip-camera** | **media-player**} • **auto qos classify** [**police**] • **auto qos trust** {**cos** | **dscp**} **Example:** `Device(config-if)# auto qos trust dscp`	The following commands enable auto-QoS for VoIP: • **auto qos voip cisco-phone**: If the port is connected to a Cisco IP Phone, the QoS labels of incoming packets are only trusted (conditional trust through CDP) when the telephone is detected. **Note** Do not configure the **auto qos voip cisco-phone** option for IP phones that support video. This option causes DSCP markings of video packets to get overwritten, because these packets do not have Expedited Forwarding priority, which results in these packets getting classified in the class-default class. • **auto qos voip cisco-softphone**: The port is connected to device running the Cisco SoftPhone feature. This command generates a QoS configuration for interfaces connected to PCs running the Cisco IP SoftPhone application and mark, as well as police traffic coming from such interfaces. Ports configured with this command are considered untrusted. • **auto qos voip trust**: The uplink port is connected to a trusted switch or router, and the VoIP traffic classification in the ingress packet is trusted. The following commands enable auto-QoS for the specified video device (system, camera, or media player): • **auto qos video cts**: A port connected to a Cisco Telepresence system. QoS labels of incoming packets are only trusted (conditional trust through CDP) when a Cisco TelePresence is detected. • **auto qos video ip-camera**: A port connected to a Cisco video surveillance

	Command or Action	Purpose
		camera. QoS labels of incoming packets are only trusted (conditional trust through CDP) when a Cisco camera is detected. • **auto qos video media-player**: A port connected to a CDP-capable Cisco digital media player. QoS labels of incoming packets are only trusted (conditional trust through CDP) when a digital media player is detected. The following command enables auto-QoS for classification: • **auto qos classify police**: This command generates a QoS configuration for untrusted interfaces. The configuration places a service-policy on the interface to classify traffic coming from untrusted desktops/devices and mark them accordingly. The service-policies generated do police. The following commands enable auto-QoS for trusted interfaces: • **auto qos trust cos**: Class of service. • **auto qos trust dscp**: Differentiated Services Code Point.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 5	**show auto qos interface** *interface-id* **Example:** `Device# show auto qos interface gigabitethernet 1/1`	(Optional) Displays the auto-QoS command on the interface on which auto-QoS was enabled. Use the **show running-config** command to display the auto-QoS configuration and user modifications.

Upgrading Auto-QoS

Before you begin

Prior to upgrading, you need to remove all auto-QoS configurations currently on the switch. This sample procedure describes that process.

After following this sample procedure, you must then reboot the switch with the new or upgraded software image and reconfigure auto-QoS.

Procedure

Step 1 **show auto qos**

Example:

```
Device# show auto qos

gigabitethernet 1/1
auto qos trust dscp
gigabitethernet 1/1
auto qos trust cos
```

In privileged EXEC mode, record all current auto QoS configurations by entering this command.

Step 2 **no auto qos**

Example:

```
Device(config-if)#no auto qos
```

In interface configuration mode, run the appropriate **no auto qos** command on each interface that has an auto QoS configuration.

Step 3 **Example:**

```
Device#
```

Return to privileged EXEC mode, and record any remaining auto QoS maps class maps, policy maps, access lists, table maps, or other configurations by entering this command.

Step 4 **no policy-map** *policy-map_name*

Example:

```
Device(config)# no policy-map pmap_101
Device(config)# no class-map cmap_101
Device(config)# no ip access-list extended AutoQos-101
Device(config)# no table-map 101
Device(config)# no table-map policed-dscp
```

In global configuration mode, remove the QoS class maps, policy maps, access-lists, table maps, and any other auto QoS configurations by entering these commands:

- **no policy-map** *policy-map-name*
- **no class-map** *class-map-name*
- **no ip access-list extended** *Auto-QoS-x*
- **no table-map** *table-map-name*

- **no table-map policed-dscp**

Step 5 **Example:**

```
Device#
```

Return to privileged EXEC mode, run this command again to ensure that no auto-QoS configuration or remaining parts of the auto-QoS configuration exists

Step 6 **show auto qos**

Example:

```
Device# show auto qos
```

Run this command to ensure that no auto-QoS configuration or remaining parts of the configuration exists.

Step 7 **write memory**

Example:

```
Device# write memory
```

Write the changes to the auto QoS configuration to NV memory by entering the **write memory** command.

What to do next

Reboot the switch with the new or upgraded software image.

After rebooting with the new or upgraded software image, re-configure auto-QoS for the appropriate switch interfaces as determined by running the **show auto qos** command described in step 1.

Note There is only one table-map for exceed and another table-map for violate markdown per switch. If the switch already has a table-map under the exceed action, then the auto-qos policy cannot be applied.

Enabling Auto-Qos Compact

To enable auto-QoS compact, enter this command:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 2	**auto qos global compact** **Example:** `Device(config)# auto qos global compact`	Enables auto-Qos compact and generates (hidden) the global configurations for auto-QoS. You can then enter the auto-QoS command you want to configure in the interface configuration mode and the interface commands that the system generates are also hidden. To display the auto-QoS configuration that has been applied, use these the privileged EXEC commands: • **show derived-config** • **show policy-map** • **show access-list** • **show class-map** • **show table-map** • **show auto qos** • **show policy-map interface** • **show ip access-lists** • **show policy-map type queue**

What to do next

To disable auto-QoS compact, remove auto-Qos instances from all interfaces by entering the **no** form of the corresponding auto-QoS commands and then enter the **no auto qos global compact** global configuration command.

Monitoring Auto-QoS

Table 128: Commands for Monitoring Auto-QoS

Command	Description
show auto qos [**interface** [*interface-id*]]	Displays the initial auto-QoS configuration. You can compare the **show auto qos** and the **show running-config** command output to identify the user-defined QoS settings.

Command	Description
show running-config	Displays information about the QoS configuration that might be affected by auto-QoS. You can compare the **show auto qos** and the **show running-config** command output to identify the user-defined QoS settings.

Troubleshooting Auto-QoS

To troubleshoot auto-QoS, use the **debug auto qos** privileged EXEC command. For more information, see the **debug auto qos** command in the command reference for this release.

To disable auto-QoS on a port, use the **no** form of the **auto qos** command interface configuration command, such as **no auto qos voip**. Only the auto-QoS-generated interface configuration commands for this port are removed. If this is the last port on which auto-QoS is enabled and you enter the **no auto qos voip** command, auto-QoS is considered disabled even though the auto-QoS-generated global configuration commands remain (to avoid disrupting traffic on other ports affected by the global configuration).

Configuration Examples for Auto-QoS

Example: auto qos trust cos

The following is an example of the **auto qos trust cos** command and the applied policies and class maps.

The following policy maps are created and applied when running this command:

- AutoQos-4.0-Trust-Cos-Input-Policy
- AutoQos-4.0-Output-Policy

The following class maps are created and applied when running this command:

- class-default (match-any)
- AutoQos-4.0-Output-Priority-Queue (match-any)
- AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
- AutoQos-4.0-Output-Trans-Data-Queue (match-any)
- AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
- AutoQos-4.0-Output-Scavenger-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)

```
Device(config)# interface gigabitethernet 1/1
Device(config-if)# auto qos trust cos
```

```
Device(config-if)# end
Device# show policy-map interface gigabitethernet 1/1

gigabitethernet 1/1

  Service-policy input: AutoQos-4.0-Trust-Cos-Input-Policy

    Class-map: class-default (match-any)
      0 packets
      Match: any
      QoS Set
         cos cos table AutoQos-4.0-Trust-Cos-Table

  Service-policy output: AutoQos-4.0-Output-Policy

    queue stats for all priority classes:
      Queueing
      priority level 1

      (total drops) 0
      (bytes output) 0

    Class-map: AutoQos-4.0-Output-Priority-Queue (match-any)
      0 packets
      Match:  dscp cs4 (32) cs5 (40) ef (46)
      Match: cos  5
      Priority: 30% (7500000 kbps), burst bytes 187500000,

      Priority Level: 1

    Class-map: AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
      0 packets
      Match:  dscp cs2 (16) cs3 (24) cs6 (48) cs7 (56)
      Match: cos  3
      Queueing

      queue-limit dscp 16 percent 80
      queue-limit dscp 24 percent 90
      queue-limit dscp 48 percent 100
      queue-limit dscp 56 percent 100
      (total drops) 0
      (bytes output) 0
      bandwidth remaining 10%

      queue-buffers ratio 10

    Class-map: AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
      0 packets
      Match:  dscp af41 (34) af42 (36) af43 (38)
      Match: cos  4
      Queueing

      (total drops) 0
      (bytes output) 0
      bandwidth remaining 10%
      queue-buffers ratio 10

    Class-map: AutoQos-4.0-Output-Trans-Data-Queue (match-any)
      0 packets
      Match:  dscp af21 (18) af22 (20) af23 (22)
      Match: cos  2
      Queueing

      (total drops) 0
```

```
    (bytes output) 0
    bandwidth remaining 10%
    queue-buffers ratio 10

  Class-map: AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
    0 packets
    Match:  dscp af11 (10) af12 (12) af13 (14)
    Match: cos  1
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 4%
    queue-buffers ratio 10

  Class-map: AutoQos-4.0-Output-Scavenger-Queue (match-any)
    0 packets
    Match:  dscp cs1 (8)
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 1%
    queue-buffers ratio 10

  Class-map: AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)
    0 packets
    Match:  dscp af31 (26) af32 (28) af33 (30)
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 10%
    queue-buffers ratio 10

  Class-map: class-default (match-any)
    0 packets
    Match: any
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 25%
    queue-buffers ratio 25
```

Example: auto qos trust dscp

The following is an example of the **auto qos trust dscp** command and the applied policies and class maps.

The following policy maps are created and applied when running this command:

- AutoQos-4.0-Trust-Dscp-Input-Policy
- AutoQos-4.0-Output-Policy

The following class maps are created and applied when running this command:

- class-default (match-any)
- AutoQos-4.0-Output-Priority-Queue (match-any)

- AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
- AutoQos-4.0-Output-Trans-Data-Queue (match-any)
- AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
- AutoQos-4.0-Output-Scavenger-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)

```
Device(config)# interface gigabitethernet 1/1
Device(config-if)# auto qos trust dscp
Device(config-if)# end
Device#show policy-map interface gigabitethernet 1/1

gigabitethernet 1/1

  Service-policy input: AutoQos-4.0-Trust-Dscp-Input-Policy

    Class-map: class-default (match-any)
      0 packets
      Match: any
      QoS Set
        dscp dscp table AutoQos-4.0-Trust-Dscp-Table

  Service-policy output: AutoQos-4.0-Output-Policy

    queue stats for all priority classes:
      Queueing
      priority level 1

      (total drops) 0
      (bytes output) 0

    Class-map: AutoQos-4.0-Output-Priority-Queue (match-any)
      0 packets
      Match:  dscp cs4 (32) cs5 (40) ef (46)
      Match: dscp  5
      Priority: 30% (30000000 kbps), burst bytes 750000000,

      Priority Level: 1

    Class-map: AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
      0 packets
      Match:  dscp cs2 (16) cs3 (24) cs6 (48) cs7 (56)
      Match: dscp  3
      Queueing
      queue-limit dscp 16 percent 80
      queue-limit dscp 24 percent 90
      queue-limit dscp 48 percent 100
      queue-limit dscp 56 percent 100
      (total drops) 0
      (bytes output) 0
      bandwidth remaining 10%

      queue-buffers ratio 10

    Class-map: AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
      0 packets
      Match:  dscp af41 (34) af42 (36) af43 (38)
```

```
  Match: dscp  4
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Trans-Data-Queue (match-any)
  0 packets
  Match:  dscp af21 (18) af22 (20) af23 (22)
  Match: dscp  2
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
  0 packets
  Match:  dscp af11 (10) af12 (12) af13 (14)
  Match: dscp  1
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 4%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Scavenger-Queue (match-any)
  0 packets
  Match:  dscp cs1 (8)
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 1%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)
  0 packets
  Match:  dscp af31 (26) af32 (28) af33 (30)
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: class-default (match-any)
  0 packets
  Match: any
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 25%
  queue-buffers ratio 25
```

Example: auto qos video cts

The following is an example of the **auto qos video cts** command and the applied policies and class maps.

The following policy maps are created and applied when running this command:

- AutoQos-4.0-Trust-Cos-Input-Policy
- AutoQos-4.0-Output-Policy

The following class maps are created and applied when running this command:

- class-default (match-any)
- AutoQos-4.0-Output-Priority-Queue (match-any)
- AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
- AutoQos-4.0-Output-Trans-Data-Queue (match-any)
- AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
- AutoQos-4.0-Output-Scavenger-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)

```
Device(config)# interface gigabitethernet 1/1
Device(config-if)# auto qos video cts
Device(config-if)# end
Device# show policy-map interface gigabitethernet 1/1
gigabitethernet 1/1

  Service-policy input: AutoQos-4.0-Trust-Cos-Input-Policy

    Class-map: class-default (match-any)
      Match: any
      QoS Set
        cos cos table AutoQos-4.0-Trust-Cos-Table

  Service-policy output: AutoQos-4.0-Output-Policy

    queue stats for all priority classes:
      Queueing
      priority level 1

      (total drops) 0
      (bytes output) 0

    Class-map: AutoQos-4.0-Output-Priority-Queue (match-any)
      Match:  dscp cs4 (32) cs5 (40) ef (46)
      Match: cos  5
      Priority: 30% (300000 kbps), burst bytes 7500000,

      Priority Level: 1

    Class-map: AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
      Match:  dscp cs3 (24) cs6 (48) cs7 (56)
      Match: cos  3
      Queueing
```

```
  queue-limit dscp 16 percent 80
  queue-limit dscp 24 percent 90
  queue-limit dscp 48 percent 100

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%

  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
  Match:  dscp af41 (34) af42 (36) af43 (38)
  Match: cos  4
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Trans-Data-Queue (match-any)
  Match:  dscp af21 (18) af22 (20) af23 (22)
  Match: cos  2
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
  Match:  dscp af11 (10) af12 (12) af13 (14)
  Match: cos  1
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 4%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Scavenger-Queue (match-any)
  Match:  dscp cs1 (8)
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 1%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)
  Match:  dscp af31 (26) af32 (28) af33 (30)
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: class-default (match-any)
  Match: any
  Queueing

  (total drops) 0
  (bytes output) 0
```

```
    bandwidth remaining 25%
    queue-buffers ratio 25
```

Example: auto qos video ip-camera

The following is an example of the **auto qos video ip-camera** command and the applied policies and class maps.

The following policy maps are created and applied when running this command:

- AutoQos-4.0-Trust-Dscp-Input-Policy
- AutoQos-4.0-Output-Policy

The following class maps are created and applied when running this command:

- class-default (match-any)
- AutoQos-4.0-Output-Priority-Queue (match-any)
- AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
- AutoQos-4.0-Output-Trans-Data-Queue (match-any)
- AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
- AutoQos-4.0-Output-Scavenger-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)

```
Device(config)# interface gigabitethernet 1/1
Device(config-if)# auto qos video ip-camera
Device(config-if)# end
Device# show policy-map interface gigabitethernet 1/1
gigabitethernet 1/1

  Service-policy input: AutoQos-4.0-Trust-Dscp-Input-Policy

    Class-map: class-default (match-any)
      Match: any
      QoS Set
        dscp dscp table AutoQos-4.0-Trust-Dscp-Table

  Service-policy output: AutoQos-4.0-Output-Policy

    queue stats for all priority classes:
      Queueing
      priority level 1

      (total drops) 0
      (bytes output) 0

    Class-map: AutoQos-4.0-Output-Priority-Queue (match-any)
      Match:  dscp cs4 (32) cs5 (40) ef (46)
      Match: cos  5
      Priority: 30% (300000 kbps), burst bytes 7500000,

      Priority Level: 1
```

```
Class-map: AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
  Match:  dscp cs3 (24) cs6 (48) cs7 (56)
  Match: cos  3
  Queueing
  queue-limit dscp 16 percent 80
  queue-limit dscp 24 percent 90
  queue-limit dscp 48 percent 100

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%

  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
  Match:  dscp af41 (34) af42 (36) af43 (38)
  Match: cos  4
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Trans-Data-Queue (match-any)
  Match:  dscp af21 (18) af22 (20) af23 (22)
  Match: cos  2
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
  Match:  dscp af11 (10) af12 (12) af13 (14)
  Match: cos  1
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 4%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Scavenger-Queue (match-any)
  Match:  dscp cs1 (8)
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 1%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)
  Match:  dscp af31 (26) af32 (28) af33 (30)
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: class-default (match-any)
```

```
    Match: any
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 25%
    queue-buffers ratio 25
```

Example: auto qos video media-player

The following is an example of the **auto qos video media-player** command and the applied policies and class maps.

The following policy maps are created and applied when running this command:

- AutoQos-4.0-Trust-Dscp-Input-Policy
- AutoQos-4.0-Output-Policy

The following class maps are created and applied when running this command:

- class-default (match-any)
- AutoQos-4.0-Output-Priority-Queue (match-any)
- AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
- AutoQos-4.0-Output-Trans-Data-Queue (match-any)
- AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
- AutoQos-4.0-Output-Scavenger-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)

```
Device(config)# interface gigabitethernet 1/1
Device(config-if)# auto qos video media-player
Device(config-if)# end
Device# show policy-map interface gigabitethernet 1/1

gigabitethernet 1/1

  Service-policy input: AutoQos-4.0-Trust-Dscp-Input-Policy

    Class-map: class-default (match-any)
      Match: any
      QoS Set
        dscp dscp table AutoQos-4.0-Trust-Dscp-Table

  Service-policy output: AutoQos-4.0-Output-Policy

    queue stats for all priority classes:
      Queueing
      priority level 1
```

```
  (total drops) 0
  (bytes output) 0

Class-map: AutoQos-4.0-Output-Priority-Queue (match-any)
  Match:  dscp cs4 (32) cs5 (40) ef (46)
  Match: cos  5
  Priority: 30% (300000 kbps), burst bytes 7500000,

  Priority Level: 1

Class-map: AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
  Match:  dscp cs3 (24) cs6 (48) cs7 (56)
  Match: cos  3
  Queueing
  queue-limit dscp 16 percent 80
  queue-limit dscp 24 percent 90
  queue-limit dscp 48 percent 100

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%

  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
  Match:  dscp af41 (34) af42 (36) af43 (38)
  Match: cos  4
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Trans-Data-Queue (match-any)
  Match:  dscp af21 (18) af22 (20) af23 (22)
  Match: cos  2
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
  Match:  dscp af11 (10) af12 (12) af13 (14)
  Match: cos  1
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 4%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Scavenger-Queue (match-any)
  Match:  dscp cs1 (8)
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 1%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)
```

```
    Match:  dscp af31 (26) af32 (28) af33 (30)
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 10%
    queue-buffers ratio 10

  Class-map: class-default (match-any)
    Match: any
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 25%
    queue-buffers ratio 25
```

Example: auto qos voip trust

The following is an example of the **auto qos voip trust** command and the applied policies and class maps.

The following policy maps are created and applied when running this command:

- AutoQos-4.0-Trust-Cos-Input-Policy
- AutoQos-4.0-Output-Policy

The following class maps are created and applied when running this command:

- class-default (match-any)
- AutoQos-4.0-Output-Priority-Queue (match-any)
- AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
- AutoQos-4.0-Output-Trans-Data-Queue (match-any)
- AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
- AutoQos-4.0-Output-Scavenger-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)

```
Device(config)# interface gigabitethernet 1/1
Device(config-if)# auto qos voip trust
Device(config-if)# end
Device# show policy-map interface gigabitethernet 1/1

gigabitethernet 1/1

  Service-policy input: AutoQos-4.0-Trust-Cos-Input-Policy

    Class-map: class-default (match-any)
      Match: any
      QoS Set
        cos cos table AutoQos-4.0-Trust-Cos-Table
```

```
Service-policy output: AutoQos-4.0-Output-Policy

  queue stats for all priority classes:
    Queueing
    priority level 1

    (total drops) 0
    (bytes output) 0

  Class-map: AutoQos-4.0-Output-Priority-Queue (match-any)
    Match:  dscp cs4 (32) cs5 (40) ef (46)
    Match: cos  5
    Priority: 30% (300000 kbps), burst bytes 7500000,

    Priority Level: 1

  Class-map: AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
    Match:  dscp cs3 (24) cs6 (48) cs7 (56)
    Match: cos  3
    Queueing
    queue-limit dscp 16 percent 80
    queue-limit dscp 24 percent 90
    queue-limit dscp 48 percent 100

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 10%

    queue-buffers ratio 10

  Class-map: AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
    Match:  dscp af41 (34) af42 (36) af43 (38)
    Match: cos  4
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 10%
    queue-buffers ratio 10

  Class-map: AutoQos-4.0-Output-Trans-Data-Queue (match-any)
    Match:  dscp af21 (18) af22 (20) af23 (22)
    Match: cos  2
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 10%
    queue-buffers ratio 10

  Class-map: AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
    Match:  dscp af11 (10) af12 (12) af13 (14)
    Match: cos  1
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 4%
    queue-buffers ratio 10

  Class-map: AutoQos-4.0-Output-Scavenger-Queue (match-any)
    Match:  dscp cs1 (8)
    Queueing
```

```
    (total drops) 0
    (bytes output) 0
    bandwidth remaining 1%
    queue-buffers ratio 10

  Class-map: AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)
    Match:  dscp af31 (26) af32 (28) af33 (30)
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 10%
    queue-buffers ratio 10

  Class-map: class-default (match-any)
    Match: any
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 25%
    queue-buffers ratio 25
```

Example: auto qos voip cisco-phone

The following is an example of the **auto qos voip cisco-phone** command and the applied policies and class maps.

The following policy maps are created and applied when running this command:

- AutoQos-4.0-CiscoPhone-Input-Policy
- AutoQos-4.0-Output-Policy

The following class maps are created and applied when running this command:

- AutoQos-4.0-Voip-Data-Class (match-any)
- AutoQos-4.0-Voip-Signal-Class (match-any)
- AutoQos-4.0-Default-Class (match-any)
- class-default (match-any)
- AutoQos-4.0-Output-Priority-Queue (match-any)
- AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
- AutoQos-4.0-Output-Trans-Data-Queue (match-any)
- AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
- AutoQos-4.0-Output-Scavenger-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)

```
Device(config)# interface gigabitethernet 1/1
```

```
Device(config-if)# auto qos voip cisco-phone
Device(config-if)# end
Device# show policy-map interface gigabitethernet 1/1

 gigabitethernet 1/1

  Service-policy input: AutoQos-4.0-CiscoPhone-Input-Policy

    Class-map: AutoQos-4.0-Voip-Data-Class (match-any)
      Match: ip dscp ef (46)
      QoS Set
        ip dscp ef
      police:
          cir 128000 bps, bc 8000 bytes, be 8000 bytes
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          set-dscp-transmit dscp table policed-dscp
        violated 0 bytes; actions:
          drop
        conformed 0000 bps, exceed 0000 bps, violate 0000 bps

    Class-map: AutoQos-4.0-Voip-Signal-Class (match-any)
      Match: ip dscp cs3 (24)
      QoS Set
        ip dscp cs3
      police:
          cir 32000 bps, bc 8000 bytes, be 8000 bytes
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          set-dscp-transmit dscp table policed-dscp
        violated 0 bytes; actions:
          drop
        conformed 0000 bps, exceed 0000 bps, violate 0000 bps

    Class-map: AutoQos-4.0-Default-Class (match-any)
      Match: access-group name AutoQos-4.0-Acl-Default
      QoS Set
        dscp default
      police:
          cir 10000000 bps, bc 8000 bytes, be 8000 bytes
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          set-dscp-transmit dscp table policed-dscp
        violated 0 bytes; actions:
          drop
        conformed 0000 bps, exceed 0000 bps, violate 0000 bps

    Class-map: class-default (match-any)
      Match: any

  Service-policy output: AutoQos-4.0-Output-Policy

    queue stats for all priority classes:
      Queueing
      priority level 1

      (total drops) 0
      (bytes output) 0

    Class-map: AutoQos-4.0-Output-Priority-Queue (match-any)
      Match:  dscp cs4 (32) cs5 (40) ef (46)
```

```
  Match: cos  5
  Priority: 30% (300000 kbps), burst bytes 7500000,

  Priority Level: 1

Class-map: AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
  Match:  dscp cs3 (24) cs6 (48) cs7 (56)
  Match: cos  3
  Queueing
  queue-limit dscp 16 percent 80
  queue-limit dscp 24 percent 90
  queue-limit dscp 48 percent 100

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%

  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
  Match:  dscp af41 (34) af42 (36) af43 (38)
  Match: cos  4
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Trans-Data-Queue (match-any)
  Match:  dscp af21 (18) af22 (20) af23 (22)
  Match: cos  2
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
  Match:  dscp af11 (10) af12 (12) af13 (14)
  Match: cos  1
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 4%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Scavenger-Queue (match-any)
  Match:  dscp cs1 (8)
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 1%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)
  Match:  dscp af31 (26) af32 (28) af33 (30)
  Queueing

  (total drops) 0
  (bytes output) 0
```

```
    bandwidth remaining 10%
    queue-buffers ratio 10

  Class-map: class-default (match-any)
    Match: any
    Queueing

    (total drops) 0
    (bytes output) 0
    bandwidth remaining 25%
    queue-buffers ratio 25
```

Example: auto qos voip cisco-softphone

The following is an example of the **auto qos voip cisco-softphone** command and the applied policies and class maps.

The following policy maps are created and applied when running this command:

- AutoQos-4.0-CiscoSoftPhone-Input-Policy
- AutoQos-4.0-Output-Policy

The following class maps are created and applied when running this command:

- AutoQos-4.0-Voip-Data-Class (match-any)
- AutoQos-4.0-Voip-Signal-Class (match-any)
- AutoQos-4.0-Multimedia-Conf-Class (match-any)
- AutoQos-4.0-Bulk-Data-Class (match-any)
- AutoQos-4.0-Transaction-Class (match-any)
- AutoQos-4.0-Scavanger-Class (match-any)
- AutoQos-4.0-Signaling-Class (match-any)
- AutoQos-4.0-Default-Class (match-any)
- class-default (match-any)
- AutoQos-4.0-Output-Priority-Queue (match-any)
- AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
- AutoQos-4.0-Output-Trans-Data-Queue (match-any)
- AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
- AutoQos-4.0-Output-Scavenger-Queue (match-any)
- AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)

```
Device(config)# interface gigabitethernet 1/1
```

```
Device(config-if)# auto qos voip cisco-softphone
Device(config-if)# end
Device# show policy-map interface gigabitethernet 1/1

 gigabitethernet 1/1

  Service-policy input: AutoQos-4.0-CiscoSoftPhone-Input-Policy

    Class-map: AutoQos-4.0-Voip-Data-Class (match-any)
      Match: ip dscp ef (46)
      QoS Set
        ip dscp ef
      police:
          cir 128000 bps, bc 8000 bytes, be 8000 bytes
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          set-dscp-transmit dscp table policed-dscp
        violated 0 bytes; actions:
          drop
        conformed 0000 bps, exceed 0000 bps, violate 0000 bps

    Class-map: AutoQos-4.0-Voip-Signal-Class (match-any)
      Match: ip dscp cs3 (24)
      QoS Set
        ip dscp cs3
      police:
          cir 32000 bps, bc 8000 bytes, be 8000 bytes
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          set-dscp-transmit dscp table policed-dscp
        violated 0 bytes; actions:
          drop
        conformed 0000 bps, exceed 0000 bps, violate 0000 bps

    Class-map: AutoQos-4.0-Multimedia-Conf-Class (match-any)
      Match: access-group name AutoQos-4.0-Acl-MultiEnhanced-Conf
      QoS Set
        dscp af41
      police:
          cir 5000000 bps, bc 8000 bytes, be 8000 bytes
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          set-dscp-transmit dscp table policed-dscp
        violated 0 bytes; actions:
          drop
        conformed 0000 bps, exceed 0000 bps, violate 0000 bps

    Class-map: AutoQos-4.0-Bulk-Data-Class (match-any)
      Match: access-group name AutoQos-4.0-Acl-Bulk-Data
      QoS Set
        dscp af11
      police:
          cir 10000000 bps, bc 8000 bytes, be 8000 bytes
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          set-dscp-transmit dscp table policed-dscp
        violated 0 bytes; actions:
          drop
        conformed 0000 bps, exceed 0000 bps, violate 0000 bps
```

```
  Class-map: AutoQos-4.0-Transaction-Class (match-any)
    Match: access-group name AutoQos-4.0-Acl-Transactional-Data
    QoS Set
      dscp af21
    police:
        cir 10000000 bps, bc 8000 bytes, be 8000 bytes
      conformed 0 bytes; actions:
        transmit
      exceeded 0 bytes; actions:
        set-dscp-transmit dscp table policed-dscp
      violated 0 bytes; actions:
        drop
      conformed 0000 bps, exceed 0000 bps, violate 0000 bps

  Class-map: AutoQos-4.0-Scavanger-Class (match-any)
    Match: access-group name AutoQos-4.0-Acl-Scavanger
    QoS Set
      dscp cs1
    police:
        cir 10000000 bps, bc 8000 bytes, be 8000 bytes
      conformed 0 bytes; actions:
        transmit
      exceeded 0 bytes; actions:
        set-dscp-transmit dscp table policed-dscp
      violated 0 bytes; actions:
        drop
      conformed 0000 bps, exceed 0000 bps, violate 0000 bps

  Class-map: AutoQos-4.0-Signaling-Class (match-any)
    Match: access-group name AutoQos-4.0-Acl-Signaling
    QoS Set
      dscp cs3
    police:
        cir 32000 bps, bc 8000 bytes, be 8000 bytes
      conformed 0 bytes; actions:
        transmit
      exceeded 0 bytes; actions:
        set-dscp-transmit dscp table policed-dscp
      violated 0 bytes; actions:
        drop
      conformed 0000 bps, exceed 0000 bps, violate 0000 bps

  Class-map: AutoQos-4.0-Default-Class (match-any)
    Match: access-group name AutoQos-4.0-Acl-Default
    QoS Set
      dscp default
    police:
        cir 10000000 bps, bc 8000 bytes, be 8000 bytes
      conformed 0 bytes; actions:
        transmit
      exceeded 0 bytes; actions:
        set-dscp-transmit dscp table policed-dscp
      violated 0 bytes; actions:
        drop
      conformed 0000 bps, exceed 0000 bps, violate 0000 bps

  Class-map: class-default (match-any)
    Match: any

Service-policy output: AutoQos-4.0-Output-Policy

  queue stats for all priority classes:
    Queueing
    priority level 1
```

```
  (total drops) 0
  (bytes output) 0

Class-map: AutoQos-4.0-Output-Priority-Queue (match-any)
  Match:  dscp cs4 (32) cs5 (40) ef (46)
  Match: cos  5
  Priority: 30% (300000 kbps), burst bytes 7500000,

  Priority Level: 1

Class-map: AutoQos-4.0-Output-Control-Mgmt-Queue (match-any)
  Match:  dscp cs3 (24) cs6 (48) cs7 (56)
  Match: cos  3
  Queueing
  queue-limit dscp 16 percent 80
  queue-limit dscp 24 percent 90
  queue-limit dscp 48 percent 100

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%

  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Multimedia-Conf-Queue (match-any)
  Match:  dscp af41 (34) af42 (36) af43 (38)
  Match: cos  4
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Trans-Data-Queue (match-any)
  Match:  dscp af21 (18) af22 (20) af23 (22)
  Match: cos  2
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Bulk-Data-Queue (match-any)
  Match:  dscp af11 (10) af12 (12) af13 (14)
  Match: cos  1
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 4%
  queue-buffers ratio 10

Class-map: AutoQos-4.0-Output-Scavenger-Queue (match-any)
  Match:  dscp cs1 (8)
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 1%
  queue-buffers ratio 10
```

```
Class-map: AutoQos-4.0-Output-Multimedia-Strm-Queue (match-any)
  Match:  dscp af31 (26) af32 (28) af33 (30)
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 10%
  queue-buffers ratio 10

Class-map: class-default (match-any)
  Match: any
  Queueing

  (total drops) 0
  (bytes output) 0
  bandwidth remaining 25%
  queue-buffers ratio 25
```

Example: auto qos global compact

The following is an example of the **auto qos global compact** command.

```
Device# configure terminal
Device(config)# auto qos global compact
Device(config)# interface gigabitethernet 1/1
Device(config-if)# auto qos voip cisco-phone

Device# show auto qos
gigabitethernet 1/1
auto qos voip cisco-phone

Device# show running-config interface gigabitethernet 1/1
interface gigabitethernet 1/1
auto qos voip cisco-phone
end
```

Where to Go Next for Auto-QoS

Review the QoS documentation if you require any specific QoS changes to your auto-QoS configuration.

CHAPTER 132

Configuring QoS

Prerequisites for QoS

Before configuring standard Quality of Service (QoS), you must have a thorough understanding of these items:

- Standard QoS concepts.
- Classic Cisco IOS QoS.
- Modular QoS CLI (MQC).
- Understanding of Virtual Output Queuing (VOQ) architecture.
- The types of applications used and the traffic patterns on your network.
- Traffic characteristics and needs of your network. For example, is the traffic on your network bursty? Do you need to reserve bandwidth for voice and video streams?
- Bandwidth requirements and speed of the network.
- Location of congestion points in the network.

Restrictions for QoS on Wired Targets

A target is an entity where a policy is applied. A wired target can be either a port or VLAN.

The following are restrictions for applying QoS features on the device for the wired target:

- A maximum of 8 queuing classes are supported on the device port for the wired target.

- A maximum of 63 policers are supported per policy on the wired port for the wired target, in the ingress or egress directions.
- A maximum of 1599 policy-maps can be created.
- The fragmented traffic is assigned to the default class **(class class-default)**.
- No more than two levels are supported in a QoS hierarchy.
- In a hierarchical policy, overlapping actions between parent and child are not allowed, except when a policy has the port shaper in the parent and queuing features in the child policy.
- A QoS policy cannot be attached to any EtherChannel interface.
- Policing in both the parent and child is not supported in a QoS hierarchy.
- Marking in both the parent and child is not supported in a QoS hierarchy.
- With shaping, there is an IPG overhead of 20Bytes for every packet that is accounted internally in the hardware. Shaping accuracy will be effected by this, specially for packets of small size.
- A maximum of 256 classes are supported per policy on the wired port for the wired target.
- Based on the Cisco UADP architecture, traffic is subjected to QoS lookup and the corresponding configured actions even if this traffic is later dropped in the Egress Global Resolution block and is never transmitted out of the actual interface.
- Only marking policy is supported on SVI.
- A port-level input marking policy takes precedence over an SVI policy; however, if no port policy is configured, the SVI policy takes precedence. For a port policy to take precedence, define a port-level policy; so that the SVI policy is overwritten.
- Classification counters have the following specific restrictions:
    - Classification counters count packets instead of bytes.
    - Filter-based classification counters are not supported
    - Only QoS configurations with marking or policing trigger the classification counter.
    - As long as there is policing or marking action in the policy, the class will have classification counters.
    - Classification counters are not supported on pure queuing policies under any class-map.
    - When there are multiple match statements in a class, the traffic counter is cumulative for all the match statements in the class.
    - Classification counters (class-map) are not available in queuing policy with actions like bandwidth, WRED, queue-buffer, shaping, and so on. The **show policy-map interface** command output will display classification counters (class-map) only for policies having either remarking or policer action.
- Egress remarking or set option is supported based on DSCP, CoS, or precedence. A new policy-map must be defined and applied on the interface in egress direction.
- The device supports a total of eight table maps for policer exceed markdown and eight table maps for policer violate markdown.
- Hierarchical policies are required for the following:

- Port-shapers
- Aggregate policers
- PV policy
- Parent shaping and child marking/policing

- For ports with wired targets, these are the only supported hierarchical policies:
  - Police chaining in the same policy is unsupported.
  - Hierarchical queuing is unsupported in the same policy (port shaper is the exception).
  - In a parent class, all filters must have the same type. The child filter type must match the parent filter type with the following exceptions:
    - If the parent class is configured to match IP, then the child class can be configured to match the ACL.
    - If the parent class is configured to match CoS, then the child class can be configured to match the ACL.

- Queuing actions are supported only on DSCP, CoS, QoS-group, IP precedence, and EXP based classification.

The following are restrictions for applying QoS features on the VLAN to the wired target:

- For a flat or nonhierarchical policy, only marking or a table map is supported.

The following are restrictions and considerations for applying QoS features on EtherChannel and channel member interfaces:

- QoS is not supported on an EtherChannel interface.
- QoS is supported on EtherChannel member interfaces in both ingress and egression directions. All EtherChannel members must have the same QoS policy applied. If the QoS policy is not the same, each individual policy on the different link acts independently.
- On attaching a service policy to channel members, the following warning message appears to remind the user to make sure the same policy is attached to all ports in the EtherChannel: ' Warning: add service policy will cause inconsistency with port xxx in ether channel xxx. '.
- Auto QoS is not supported on EtherChannel members.

Note On attaching a service policy to an EtherChannel, the following message appears on the console: ' Warning: add service policy will cause inconsistency with port xxx in ether channel xxx. '. This warning message should be expected. This warning message is a reminder to attach the same policy to other ports in the same EtherChannel. The same message will be seen during boot up. This message does not mean there is a discrepancy between the EtherChannel member ports.

Information About QoS

The following sections provide information about QoS.

QoS Components

Quality of service (QoS) consists of the following key components:

- Classification: Classification is the process of distinguishing one type of traffic from another based upon access control lists (ACLs), Differentiated Services Code Point (DSCP), Class of Service (CoS), and other factors.
- Marking and mutation: Marking is used on traffic to convey specific information to a downstream device in the network, or to carry information from one interface in a device to another. When traffic is marked, QoS operations on that traffic can be applied. This can be accomplished directly using the **set** command or through a table map, which takes input values and translates them directly to values on output.
- Shaping and policing: Shaping is the process of imposing a maximum rate of traffic, while regulating the traffic rate in such a way that downstream devices are not subjected to congestion. Shaping in the most common form is used to limit the traffic sent from a physical or logical interface. Policing is used to impose a maximum rate on a traffic class. If the rate is exceeded, then a specific action is taken as soon as the event occurs.
- Queuing: Queuing is used to prevent traffic congestion. Traffic is sent to specific queues for servicing and scheduling based upon bandwidth allocation. Traffic is then scheduled or sent out through the port.
- Bandwidth: Bandwidth allocation determines the available capacity for traffic that is subject to QoS policies.
- Trust: Trust enables traffic to pass through the device, and the Differentiated Services Code Point (DSCP), precedence, or CoS values coming in from the end points are retained in the absence of any explicit policy configuration.

QoS Terminology

The following terms are used interchangeably in this QoS configuration guide:

- Upstream (direction towards the device) is the same as ingress.
- Downstream (direction from the device) is the same as egress.

Information About QoS

By configuring the quality of service (QoS), you can provide preferential treatment to specific types of traffic at the expense of other traffic types. Without QoS, the device offers best-effort service for each packet, regardless of the packet contents or size. The device sends the packets without any assurance of reliability, delay bounds, or throughput.

The following are specific features provided by QoS:

- Low latency

- Bandwidth guarantee
- Buffering capabilities and dropping disciplines
- Traffic policing
- Enables the changing of the attribute of the frame or packet header
- Relative services

Modular QoS CLI

QoS features are enabled through the Modular QoS CLI (MQC). The MQC is a CLI structure that allows you to create traffic policies and attach these policies to interfaces. A traffic policy contains a traffic class and one or more QoS features. A traffic class is used to classify traffic, while the QoS features in the traffic policy determine how to treat the classified traffic. One of the main goals of MQC is to provide a platform-independent interface for configuring QoS across Cisco platforms.

QoS Wired Access Features

The following table describes the supported QoS features for wired access.

Table 129: QoS Wired Access Features

Feature	Description
Supported targets	• Gigabit Ethernet • Multigigabit Ethernet • 10-Gigabit Ethernet • 25-Gigabit Ethernet • VLAN
Configuration sequence	QoS policy installed using the **service-policy** command.
Supported number of queues at port level	Up to eight queues supported on a port.
Supported classification mechanism	• DSCP • IP precedence • CoS • QoS-group • ACL membership including: • IPv4 ACLs • IPv6 ACLS • MAC ACLs

Hierarchical QoS

The device supports hierarchical QoS (HQoS). HQoS allows you to perform:

- Hierarchical classification: Traffic classification is based upon other classes.
- Hierarchical policing: The process of having the policing configuration at multiple levels in a hierarchical policy.
- Hierarchical shaping: Shaping can also be configured at multiple levels in the hierarchy.

Note Hierarchical shaping is only supported for the port shaper, where for the parent you only have a configuration for the class default, and the only action for the class default is shaping.

QoS Implementation

Typically, networks operate on a best-effort delivery basis, which means that all traffic has equal priority and an equal chance of being delivered in a timely manner. When congestion occurs, all traffic has an equal chance of being dropped.

When you configure the QoS feature, you can select specific network traffic, prioritize it according to its relative importance, and use congestion-management and congestion-avoidance techniques to provide preferential treatment. Implementing QoS in your network makes network performance more predictable and bandwidth utilization more effective.

QoS implementation is based on the Differentiated Services (Diff-Serv) architecture, a standard from the Internet Engineering Task Force (IETF). This architecture specifies that each packet is classified upon entry into the network.

The classification is carried in the IP packet header, using six bits from the deprecated IP type of service (ToS) field to carry the classification (*class*) information. Classification can also be carried in the Layer 2 frame.

Layer 2 Frame Prioritization Bits

Layer 2 Inter-Switch Link (ISL) frame headers have a 1-byte User field that carries an IEEE 802.1p class of service (CoS) value in the three least-significant bits. On ports configured as Layer 2 ISL trunks, all the traffic is in ISL frames.

Layer 2 802.1Q frame headers have a 2-byte Tag Control Information field that carries the CoS value in the three most-significant bits, which are called User Priority bits. On ports configured as Layer 2 802.1Q trunks, all the traffic is in 802.1Q frames, except for the traffic in the native VLAN.

Other frame types cannot carry Layer 2 CoS values.

Layer 2 CoS values range from 0 for low priority to 7 for high priority.

Layer 3 Packet Prioritization Bits

Layer 3 IP packets can carry either an IP precedence value or a Differentiated Services Code Point (DSCP) value. QoS supports the use of either value because DSCP values are backward-compatible with IP precedence values.

IP precedence values range from 0 to 7. DSCP values range from 0 to 63.

End-to-End QoS Solution Using Classification

All the switches and the routers that access the internet rely on class information to provide the same forwarding treatment to packets with the same class information, and different treatment to packets with different class information. The class information in the packet can be assigned by end hosts or by switches or routers along the way, based on a configured policy, detailed examination of the packet, or both. Detailed examination of the packet is expected to occur closer to the edge of the network, so that the core switches and routers are not overloaded with this task.

Switches and routers along the path can use the class information to limit the amount of resources allocated per traffic class. The behavior of an individual device when handling traffic in the Diff-Serv architecture is called per-hop behavior. If all the devices along a path provide a consistent per-hop behavior, you can construct an end-to-end QoS solution.

Implementing QoS in your network can be a simple task or a complex task, depending on the QoS features offered by your internetworking devices, the traffic types and patterns in your network, and the granularity of control that you need over incoming and outgoing traffic.

Packet Classification

Packet classification is the process of identifying a packet as belonging to one of several classes in a defined policy, based on certain criteria. The Modular QoS CLI (MQC) is a policy-class based language. The policy class language is used to define the following:

- Class map template with one or several match criteria
- Policy map template with one or several classes associated to the policy map

The policy map template is then associated to one or several interfaces on the device.

Packet classification is the process of identifying a packet as belonging to one of the classes defined in the policy map. The process of classification will exit when the packet being processed matches a specific filter in a class. This is referred to as first-match exit. If a packet matches multiple classes in a policy, irrespective of the order of classes in the policy map, it will still exit the classification process after matching the first class.

If a packet does not match any of the classes in the policy, it is classified into the default class in the policy. Every policy map has a default class, which is a system-defined class to match the packets that do not match any of the user-defined classes.

Packet classification can be categorized into the following types:

- Classification based on information that is propagated with the packet
- Classification based on information that is device specific
- Hierarchical classification

Classification Based on Information Propagated with a Packet

Classification based on information that is part of a packet, and propagated either end-to-end or between hops, typically includes the following:

- Classification based on Layer 3 or 4 headers

- Classification based on Layer 2 information

Classification Based on Layer 3 or Layer 4 Header

This is the most common deployment scenario. Numerous fields in the Layer 3 and Layer 4 headers can be used for packet classification.

At the most granular level, this classification methodology can be used to match an entire flow. For this deployment type, an access control list (ACL) can be used. ACLs can also be used based on various subsets of the flow, for example, source IP address only, destination IP address only, or a combination of both.

Classification can also be done based on the precedence or DSCP values in the IP header. The IP precedence field is used to indicate the relative priority with which a particular packet needs to be handled. It is made up of three bits in the IP header's type of service (ToS) byte.

The following table shows the different IP precedence bit values and their names.

Table 130: IP Precedence Values and Names

IP Precedence Value	IP Precedence Bits	IP Precedence Names
0	000	Routine
1	001	Priority
2	010	Immediate
3	011	Flash
4	100	Flash Override
5	101	Critical
6	110	Internetwork control
7	111	Network control

Note All the routing control traffic in a network use the IP precedence value 6 by default. IP precedence value 7 is also reserved for network control traffic. Therefore, we do not recommend the use of IP precedence values 6 and 7 for user traffic.

The DSCP field is made up of 6 bits in the IP header, and is being standardized by the Internet Engineering Task Force (IETF) Differentiated Services Working Group. The original ToS byte, which contained the DSCP bits, has been renamed the DSCP byte. The DSCP field is part of the IP header, similar to IP precedence. The DSCP field is a super set of the IP precedence field. Therefore, the DSCP field is used and is set in ways that are similar to what was described with respect to IP precedence.

Note

- The DSCP field definition is backward-compatible with the IP precedence values.
- Some fields in the Layer 2 header can also be set using a policy.

Classification Based on Layer 2 Header

A variety of methods can be used to perform classification based on the Layer 2 header information. The most common methods include the following:

- MAC address-based classification (only for access groups): Classification is based upon the source MAC address (for policies in the input direction) and destination MAC address (for policies in the output direction).
- Class-of-Service: Classification is based on the 3 bits in the Layer 2 header based on the IEEE 802.1p standard. This usually maps to the ToS byte in the IP header.
- VLAN ID: Classification is based on the VLAN ID of the packet.

Note Some of these fields in the Layer 2 header can also be set using a policy.

Classification Based on Information that is Device Specific

A device also provides classification mechanisms in scenarios that are available where classification is not based on information in the packet header or payload.

At times you might be required to aggregate traffic coming from multiple input interfaces into a specific class in the output interface. For example, multiple customer edge routers might be going into the same access device on different interfaces. The service provider might want to police all the aggregate voice traffic going into the core to a specific rate. However, the voice traffic coming in from the different customers could have different ToS settings. QoS group-based classification is a feature that is useful in these scenarios.

Policies configured on the input interfaces set the QoS group to a specific value, which can then be used to classify the packets in the policies enabled on the output interface.

The QoS group is a field in the packet data structure that is internal to a device. It is important to note that a QoS group is an internal label to the device and is not a part of the packet header.

QoS Wired Model

To implement QoS, the device must perform the following tasks:

- Traffic classification: Distinguish packets or flows from one another.
- Traffic marking and policing: Assign a label to indicate the given quality of service as the packets move through the device, and then make the packets comply with the configured resource usage limits.
- Queuing and scheduling: Provide different treatment in all the situations where resource contention exists.
- Shaping: Ensure that the traffic sent from the device meets a specific traffic profile.

Ingress Port Activity

The following activities occur at the ingress port of a device:

- Classification: Classifying a distinct path for a packet by associating it with a QoS label. For example, the device maps the CoS or DSCP in the packet to a QoS label to distinguish one type of traffic from another. The QoS label that is generated identifies all future QoS actions to be performed on this packet.

- Policing: Policing determines whether a packet is in or out of profile by comparing the rate of the incoming traffic to the configured policer. The policer limits the bandwidth consumed by a flow of traffic. The result is passed to the marker.
- Marking: Marking evaluates the policer and configuration information for the action to be taken when a packet is out of profile and determines what to do with the packet (pass through a packet without modification, mark down the QoS label in the packet, or drop the packet).

Egress Port Activity

The following activities occur at the egress port of the device:

- Policing—Policing determines whether a packet is in or out of profile by comparing the rate of the incoming traffic to the configured policer. The policer limits the bandwidth consumed by a flow of traffic. The result is passed to the marker.
- Marking—Marking evaluates the policer and configuration information for the action to be taken when a packet is out of profile and determines what to do with the packet (pass through a packet without modification, mark down the QoS label in the packet, or drop the packet).
- Queuing—Queuing evaluates the QoS packet label and the corresponding DSCP or CoS value before selecting which of the egress queues to use. Because congestion can occur when multiple ingress ports simultaneously send data to an egress port, Weighted Tail Drop (WTD) differentiates traffic classes and subjects the packets to different thresholds based on the QoS label. If the threshold is exceeded, the packet is dropped.

Classification

Classification is the process of distinguishing one kind of traffic from another by examining the fields in the packet. Classification is enabled only if QoS is enabled on a device. By default, QoS is enabled in a device.

During classification, a device performs a lookup and assigns a QoS label to a packet. The QoS label identifies all the QoS actions to be performed on a packet and from which queue the packet is sent.

Access Control Lists

You can use IP standard, IP extended, or Layer 2 MAC ACLs to define a group of packets with the same characteristics (class). You can also classify IP traffic based on IPv6 ACLs.

In the QoS context, the permit and deny actions in the access control entries (ACEs) have different meanings from security ACLs:

- If a match with a permit action is encountered (first-match principle), the specified QoS-related action is taken.
- If a match with a deny action is encountered, the ACL being processed is skipped, and the next ACL is processed.
- If no match with a permit action is encountered and all the ACEs have been examined, no QoS processing occurs on the packet, and the device offers best-effort service to the packet.
- If multiple ACLs are configured on a port, the lookup stops after the packet matches the first ACL with a permit action, and QoS processing begins.

Note When creating an access list, note that by default the end of the access list contains an implicit deny statement for everything if it did not find a match before reaching the end.

After a traffic class has been defined with the ACL, you can attach a policy to it. A policy might contain multiple classes with actions specified for each one of them. A policy might include commands to classify the class as a particular aggregate (for example, assign a DSCP) or rate-limit the class. This policy is then attached to a particular port on which it becomes effective.

You implement IP ACLs to classify IP traffic by using the **access-list** global configuration command; you implement Layer 2 MAC ACLs to classify non-IP traffic by using the **mac access-list extended** global configuration command.

Class Maps

A class map is a mechanism that you use to name a specific traffic flow (or class) and isolate it from all other traffic. The class map defines the criteria used to match against a specific traffic flow to further classify it. The criteria can include matching the access group defined by the ACL or matching a specific list of DSCP or IP precedence values or CoS values. If you have more than one type of traffic that you want to classify, you can create another class map and use a different name. After a packet is matched against the class-map criteria, you further classify it through the use of a policy map.

Note You cannot configure IPv4 and IPv6 classification criteria simultaneously in the same class-map. However, they can be configured in different class-maps in the same policy.

You create a class map by using the **class-map** global configuration command or the **class** policy-map configuration command. You should use the **class-map** command when the map is shared among mulitple policies. When you enter the **class-map** command, the device enters the class-map configuration mode.

You can create a default class by using the **class class-default** policy-map configuration command. The default class is system-defined and cannot be configured. Unclassified traffic (traffic that does not meet the match criteria specified in the traffic classes) is treated as default traffic.

Layer 3 Packet Length Classification

This feature provides the capability of matching and classifying traffic on the basis of the Layer 3 packet length in the IP header. The Layer 3 packet length is the IP datagram length plus the IP header length. You can set the packet length as a matching criterion in the class policy-map, to match the value on the incoming packet. The classified packet is either marked or policed based on the policy-map action. This feature does not work on IPv6 packets.

The following is an example of Layer 3 packet length classification:

```
Service-policy output: PACKET_MATCH1

Class-map: class-default (match-any)
 16281588 packets
  Match: any

Service-policy : L3_MATCH
```

```
Class-map: PACKET_LENGTH_1 (match-any)
  9910510 packets
  Match: packet length 7582
  Match: packet length 5000
  QoS Set
  dscp cs2
  police:
 rate 3 %
 rate 1200000000 bps, burst 37500000 bytes
 conformed 10000 bytes; actions:
   transmit
 exceeded 112121 bytes; actions:
   drop
 conformed 500 bps, exceeded 3434 bps

Class-map: PACKET_LENGTH_2 (match-all)
 6371042 packets
  Match: dscp cs4 (32)
  Match: packet length 7759
  police:
 rate 12000000000 bps, burst 375000000 bytes
 conformed 44545 bytes; actions:
 transmit
 exceeded 34343 bytes; actions:
 drop
 conformed 1211 bps, exceeded 11211 bps

Class-map: class-default (match-any)
 36 packets
  Match: any
 QoS Set
   precedence 3
Device#

class-map match-any PACKET_LENGTH_1
match packet length min 7582 max 7582
match packet length min 5000 max 5000

class-map match-all PACKET_LENGTH_2
match dscp cs4
match packet length min 7759 max 7759
```

Policy Maps

A policy map specifies which traffic class to act on. Actions can include the following:

- Setting a specific DSCP or IP precedence value in the traffic class
- Setting a CoS value in the traffic class
- Setting a QoS group
- Specifying the traffic bandwidth limitations and the action to take when the traffic is out of profile

Before a policy map can be effective, you must attach it to a port.

You create and name a policy map using the **policy-map** global configuration command. When you enter this command, the device enters the policy-map configuration mode. In this mode, you specify the actions to take on a specific traffic class by using the **class** or **set** policy-map configuration and policy-map class configuration commands.

The policy map can also be configured using the **police** and **bandwidth** policy-map class configuration commands, which define the policer, the bandwidth limitations of the traffic, and the action to take if the limits are exceeded. In addition, the policy-map can further be configured using the **priority** policy-map class configuration command, to schedule priority for the class or the queuing policy-map class configuration commands, **queue-buffers** and **queue-limit**.

To enable the policy map, you attach it to a port by using the **service-policy** interface configuration command.

Policy Map on Physical Port

You can configure a nonhierarchical policy map on a physical port that specifies which traffic class to act on. Actions can include setting a specific DSCP or IP precedence or CoS values in the traffic class, specifying the traffic bandwidth limitations for each matched traffic class (policer), and taking action when the traffic is out of profile (marking).

A policy map also has these characteristics:

- A policy map can contain multiple class statements, each with different match criteria and policers.
- A policy map can contain a predefined default traffic class explicitly placed at the end of the map.

  When you configure a default traffic class by using the **class class-default** policy-map configuration command, unclassified traffic (traffic that does not meet the match criteria specified in the traffic classes) is treated as the default traffic class (**class-default**).
- A separate policy-map class can exist for each type of traffic received through a port.

Policy Map on VLAN

The device supports a VLAN QoS feature that allows the user to perform QoS treatment at the VLAN level (classification and QoS actions) using the incoming frame's VLAN information. In VLAN-based QoS, a service policy is applied to an SVI interface. All physical interfaces belonging to a VLAN policy map then need to be programmed to refer to the VLAN-based policy maps instead of the port-based policy map.

Although the policy map is applied to the VLAN SVI, only marking or remarking actions can be performed on a per-port basis. You cannot configure the policer to take account of the sum of traffic from a number of physical ports. Each port needs to have a separate policer governing the traffic coming into that port.

QoS Profile

The device uses Ternary Content Addressable Memory (TCAM) to store classification rules. To optimize the usage of TCAM resources, use the QoS profile to turn off some of the lesser used features and turn them on when required.

With the **qos profile** {**default** | **extended**} command, you can select the required classification feature set. **default** keyword loads only the common classification features. **extended** keyword loads the complete classification feature set (but with a reduced scale) that are available for the device. By default, only the commonly used classification features are set on the device.

qos profile extended enables TCP flag along with the common classification features.

You can verify the QoS profile that is configured on the device using the **show platform software fed active qos profile** command.

Example

```
device# show platform software fed active qos profile
Using default - Common Classification Features
```

Security Group Classification

Security Group classification includes both source and destination groups, which are specified by source security group tag (SGT) and destination security group tag (DGT) respectively.

The objective of SGT QoS classification is to leverage user groups to increase policy granularity such that the policy isn't only application-aware but also provides some level of differentiated service based on the user identity (or the group of users to which they belong).

Egress QoS classification based on SGT or DGT isn't supported.

SGT Based QoS

The SGT based QoS feature provides a special treatment for a class of traffic that is based on the QoS policies and actions, for a defined user group or device. This feature enables you to assign multiple QoS policies to an application or traffic type that is initiated by different user groups. Each user group is defined by a unique SGT value and can support MQC-based QoS configuration.

The SGT based QoS feature is applicable to both the user group and the device-based QoS service levels for SGT-DGT-based packet classification. It can also potentially support defining of user groups based on contextual information for QoS policy prioritization.

Sharing DGID with SGACL

Due to resource limitations, only 4096 security group destination tags (DGTs) are supported. Classification based on DGT is achieved through a security destination tag ID known as DGID. DGID is a global resource and is shared with SGACL. DGID allocation is done on a first-come-first-serve basis. On a device, at startup, SGACL configuration is applied before QoS policy configuration. Hence DGID is first allocated for SGACL and then for QoS policy.

The **show platform software fed sw active sgacl detail** command displays the DGT to DGID mapping.

Example

```
device# show platform software fed active sgacl detail

Global Enforcement: On
*Refcnt: for the non-SGACL feature
========================== DGID Table ====================================
SGT/Refcnt     DGT     DGID   hash  test_cell monitor   permitted  denied
==========================================================================
*/1            24        1     24
 24            24        1     24       Off     Off        0           0
```

Restrictions for SGT Based QoS

The following are the limitations of the SGT based QoS feature:

- SGT based QoS is not supported on tunnel interfaces.
- Only 4096 security destination tags and 65539 security source tags are supported.
- SGT based policy can only be attached to the input direction of an interface.

Restrictions for an Upgrade or Downgrade

- For an upgrade from an earlier release, the maximum supported DGID is 256. Reload the switch to overcome this issue.
- For a downgrade to a previous release, the allocated DGID is displayed as 4096; but only 256 DGIDs are supported. Reload the switch to overcome this issue.
- An In-Service Software Upgrade (ISSU) fails if the tcp flag is set in a policy. To do an ISSU, first remove the tcp flag configuration.
- If a policy-map, which is attached to an interface, classifies traffic based on tcp flag, an ISSU upgrade fails. To do an ISSU, either detach the policy-map from the interface or remove the tcp flag classification.

Policing

After a packet is classified and has a DSCP-based, CoS-based, or QoS-group label assigned to it, the policing and marking process can begin.

Policing involves creating a policer that specifies the bandwidth limits for the traffic. Packets that exceed the limits are *out of profile* or *nonconforming*. Each policer decides on a packet-by-packet basis whether the packet is in or out of profile and specifies the actions on the packet. These actions, carried out by the marker, include passing through the packet without modification, dropping the packet, or modifying (marking down) the assigned DSCP or CoS value of the packet and allowing the packet to pass through.

To avoid out-of-order packets, both conform and nonconforming traffic typically exit the same queue.

Note All traffic, regardless of whether it is bridged or routed, is subjected to a policer, if one is configured. As a result, bridged packets might be dropped or might have their DSCP or CoS fields modified when they are policed and marked.

You can only configure policing on a physical port.

After you configure the policy map and policing actions, attach the policy-map to an ingress or egress port by using the **service-policy** interface configuration command.

Token-Bucket Algorithm

Policing uses a token-bucket algorithm. As each frame is received by the device, a token is added to the bucket. The bucket has a hole in it and leaks at a rate that you specify as the average traffic rate in bits per second. Each time a token is added to the bucket, the device verifies that there is enough room in the bucket. If there is not enough room, the packet is marked as nonconforming, and the specified policer action is taken (dropped or marked down).

How quickly the bucket fills is a function of the bucket depth (burst-byte), the rate at which the tokens are removed (rate-bps), and the duration of the burst above the average rate. The size of the bucket imposes an upper limit on the burst length and limits the number of frames that can be transmitted back-to-back. If the burst is short, the bucket does not overflow, and no action is taken against the traffic flow. However, if a burst is long and at a higher rate, the bucket overflows, and the policing actions are taken against the frames in that burst.

You configure the bucket depth (the maximum burst that is tolerated before the bucket overflows) by using the burst-byte option of the **police** policy-map class configuration command. You configure how fast (the

average rate) that the tokens are removed from the bucket by using the rate option of the **police** policy-map class configuration command.

Marking

Marking is used to convey specific information to a downstream device in the network, or to carry information from one interface in a device to another.

Marking can be used to set certain field/bits in the packet headers, or marking can also be used to set certain fields in the packet structure that is internal to the device. Additionally, the marking feature can be used to define mapping between fields. The following marking methods are available for QoS:

- Packet header
- Device specific information
- Table maps

Packet Header Marking

Marking on fields in the packet header can be classified into two general categories:

- IPv4/v6 header bit marking
- Layer 2 header bit marking

The marking feature at the IP level is used to set the precedence or the DSCP in the IP header to a specific value to get a specific per-hop behavior at the downstream device (switch or router), or it can also be used to aggregate traffic from different input interfaces into a single class in the output interface. The functionality is currently supported on both the IPv4 and IPv6 headers.

Marking in the Layer 2 headers is typically used to influence dropping behavior in the downstream devices (switch or router). It works in tandem with the match on the Layer 2 headers. The bits in the Layer 2 header that can be set using a policy map are class of service.

Switch-Specific Information Marking

This form of marking includes marking of fields in the packet data structure that are not part of the packets header, so that the marking can be used later in the data path. This is not propagated between the switches. Marking of QoS group falls into this category. This form of marking is only supported in policies that are enabled on the input interfaces. The corresponding matching mechanism can be enabled on the output interfaces on the same switch and an appropriate QoS action can be applied.

Table Map Marking

Table map marking enables the mapping and conversion from one field to another using a conversion table. This conversion table is called a table map.

Depending upon the table map attached to an interface, CoS, DSCP, and Precedence values of the packet are rewritten. The device allows configuring both ingress table map policies and egress table map policies.

As an example, a table map can be used to map the Layer 2 CoS setting to a precedence value in Layer 3. This feature enables combining multiple **set** commands into a single table, which indicates the method to perform the mapping. This table can be referenced in multiple policies, or multiple times in the same policy.

A table map-based policy supports the following capabilities:

- Mutation: You can have a table map that maps from one DSCP value set to another DSCP value set, and this can be attached to an egress port.
- Rewrite: Packets coming in are rewritten depending upon the configured table map.
- Mapping: Table map based policies can be used instead of set policies.

The following steps are required for table map marking:

1. Define the table map: Use the **table-map** global configuration command to map the values. The table does not know of the policies or classes within which it will be used. The default command in the table map is used to indicate the value to be copied into the to field when there is no matching from field.
2. Define the policy map: You must define the policy map where the table map will be used.
3. Associate the policy to an interface.

Note A table map policy on an input port changes the trust setting of that port to the from type of qos-marking.

Note In order to trust a value other than the dscp value, use table map with default copy in the ingress direction.

Note When you map a QoS group to a DSCP value in an egress table map policy, the QoS group does not map the equivalent COS value for DSCP. Configure a separate QoS group to COS table map if you want to define the QoS group to a non-zero COS value.

Traffic Conditioning

To support QoS in a network, traffic entering the service provider network needs to be policed on the network boundary routers to ensure that the traffic rate stays within the service limit. Even if a few routers at the network boundary start sending more traffic than what the network core is provisioned to handle, the increased traffic load leads to network congestion. The degraded performance in the network makes it difficult to deliver QoS for all the network traffic.

Traffic policing functions (using the police feature) and shaping functions (using the traffic shaping feature) manage the traffic rate, but differ in how they treat traffic when tokens are exhausted. The concept of tokens comes from the token bucket scheme, a traffic metering function.

Note When running QoS tests on network traffic, you may see different results for the shaper and policing data. Network traffic data from shaping provides more accurate results.

This table compares the policing and shaping functions.

Table 131: Comparison Between Policing and Shaping Functions

Policing Function	Shaping Function
Sends conforming traffic up to the line rate and allows bursts.	Smooths traffic and sends it out at a constant rate.
When tokens are exhausted, action is taken immediately.	When tokens are exhausted, it buffers packets and sends them out later, when tokens are available. A class with shaping has a queue associated with it which will be used to buffer the packets.
Policing has multiple units of configuration – in bits per second, packets per second and cells per second.	Shaping has only one unit of configuration - in bits per second.
Policing has multiple possible actions associated with an event, marking and dropping being example of such actions.	Shaping does not have the provision to mark packets that do not meet the profile.
Works for both input and output traffic.	Implemented for output traffic only.
Transmission Control Protocol (TCP) detects the line at line speed but adapts to the configured rate when a packet drop occurs by lowering its window size.	TCP can detect that it has a lower speed line and adapt its retransmission timer accordingly. This results in less scope of retransmissions and is TCP-friendly.

Policing

The QoS policing feature is used to impose a maximum rate on a traffic class. The QoS policing feature can also be used with the priority feature to restrict priority traffic. If the rate is exceeded, then a specific action is taken as soon as the event occurs. The rate (committed information rate [CIR] and peak information rate [PIR]) and the burst parameters (conformed burst size [B_c] and extended burst size [B_e]) are all configured in bytes per second.

The following policing forms or policers are supported for QoS:

- Single-rate two-color policing
- Dual-rate three-color policing

Note Single-rate three-color policing is not supported.

Single-Rate Two-Color Policing

Single-rate two-color policer is the mode in which you configure only a CIR and a B_c.

The B_c is an optional parameter, and if it is not specified it is computed by default. In this mode, when an incoming packet has enough tokens available, the packet is considered to be conforming. If at the time of packet arrival, enough tokens are not available within the bounds of B_c, the packet is considered to have exceeded the configured rate.

Note For information about the token-bucket algorithm, see Token-Bucket Algorithm, on page 1895.

Dual-Rate Three-Color Policing

With the dual rate policer, the device supports only color-blind mode. In this mode, you configure a committed information rate (CIR) and a peak information rate (PIR). As the name suggests, there are two token buckets in this case, one for the peak rate, and one for the conformed rate.

Note For information about the token-bucket algorithm, see Token-Bucket Algorithm, on page 1895.

In the color-blind mode, the incoming packet is first checked against the peak rate bucket. If there are not enough tokens available, the packet is said to violate the rate. If there are enough tokens available, then the tokens in the conformed rate buckets are checked to determine if there are enough tokens available. The tokens in the peak rate bucket are decremented by the size of the packet. If it does not have enough tokens available, the packet is said to have exceeded the configured rate. If there are enough tokens available, then the packet is said to conform, and the tokens in both the buckets are decremented by the size of the packet.

The rate at which tokens are replenished depends on the packet arrival. Assume that a packet comes in at time T1 and the next one comes in at time T2. The time interval between T1 and T2 determines the number of tokens that need to be added to the token bucket. This is calculated as:

Time interval between packets (T2-T1) * CIR)/8 bytes

Shaping

Shaping is the process of imposing a maximum rate of traffic, while regulating the traffic rate in such a way that the downstream switches and routers are not subjected to congestion. Shaping in the most common form is used to limit the traffic sent from a physical or logical interface.

Shaping has a buffer associated with it that ensures that packets which do not have enough tokens are buffered as opposed to being immediately dropped. The number of buffers available to the subset of traffic being shaped is limited and is computed based on a variety of factors. The number of buffers available can also be tuned using specific QoS commands. Packets are buffered as buffers are available, beyond which they are dropped.

Class-Based Traffic Shaping

The device uses class-based traffic shaping. This shaping feature is enabled on a class in a policy that is associated to an interface. A class that has shaping configured is allocated a number of buffers to hold the packets that do not have tokens. The buffered packets are sent out from the class using FIFO. In the most common form of usage, class-based shaping is used to impose a maximum rate for an physical interface or logical interface as a whole. The following shaping forms are supported in a class:

- Average rate shaping
- Hierarchical shaping

Shaping is implemented using a token bucket. The values of CIR, B_c and B_e determine the rate at which the packets are sent out and the rate at which the tokens are replenished.

Note For information about the token-bucket algorithm, see Token-Bucket Algorithm, on page 1895.

Average Rate Shaping

You use the **shape average** policy-map class command to configure average rate shaping.

This command configures a maximum bandwidth for a particular class. The queue bandwidth is restricted to this value even though the port has more bandwidth available. The device supports configuring shape average by either a percentage or by a target bit rate value.

Hierarchical Shaping

Shaping can also be configured at multiple levels in a hierarchy. This is accomplished by creating a parent policy with shaping configured, and then attaching child policies with additional shaping configurations to the parent policy.

The port shaper uses the class default and the only action permitted in the parent is shaping. The queuing action is in the child with the port shaper. With the user configured shaping, you cannot have queuing action in the child.

Queuing and Scheduling

The device uses both queuing and scheduling to help prevent traffic congestion. The device supports the following queuing and scheduling features:

- Bandwidth
- Weighted Tail Drop
- Priority queues
- Queue buffers
- Weighted Random Early Detection

When you define a queuing policy on a port, control packets are mapped to the best priority queue with the highest threshold. Control packets queue mapping works differently in the following scenarios:

- Without a quality of service (QoS) policy: If no QoS policy is configured, control packets with DSCP values 16, 24, 48, and 56 are mapped to queue 0 with the highest threshold of threshold2.
- With an user-defined policy: An user-defined queuing policy configured on egress ports can affect the default priority queue setting on control packets.

Note Queuing policy in egress direction does not support **match access-group** classification.

Control traffic is redirected to the best queue based on the following rules:

1. If defined in a user policy, the highest- level priority queue is always chosen as the best queue.

2. In the absence of a priority queue, Cisco IOS software selects queue 0 as the best queue. When the software selects queue 0 as the best queue, you must define the highest bandwidth to this queue to get the best QoS treatment to the control plane traffic.
3. If thresholds are not configured on the best queue, Cisco IOS software assigns control packets with Differentiated Services Code Point (DSCP) values 16, 24, 48, and 56 are mapped to threshold2 and reassigns the rest of the control traffic in the best queue to threshold1.

If a policy is not configured explicitly for control traffic, the Cisco IOS software maps all unmatched control traffic to the best queue with threshold2, and the matched control traffic is mapped to the queue as configured in the policy.

Note To provide proper QoS for Layer 3 packets, you must ensure that packets are explicitly classified into appropriate queues. When the software detects DSCP values in the default queue, then it automatically reassigns this queue as the best queue.

Bandwidth

The device supports the following bandwidth configurations:

- Bandwidth
- Bandwidth percent
- Bandwidth remaining precent

Bandwidth Percent

You can use the **bandwidth percent** policy-map class command to allocate a minimum bandwidth to a particular class. The total sum cannot exceed 100 percent and in case the total sum is less than 100 percent, then the rest of the bandwidth is divided equally among all bandwidth queues.

Note A queue can oversubscribe bandwidth in case the other queues do not utilize the entire port bandwidth.

You cannot mix bandwidth types on a policy map. For example, you cannot configure bandwidth in a single policy map using both a bandwidth percent and in kilobits per second.

Bandwidth Remaining Percent

Use the **bandwidth remaining percent** policy-map class command to create a percent for sharing unused bandwidth in specified queues. Any unused bandwidth will be used by these specific queues in the percent that is specified by the configuration. Use this command when the **priority** command is also used for certain queues in the policy.

When you assign percent, the queues will be assigned certain weights which are inline with these percents.

You can specify a percent between 0 to 100. For example, you can configure a bandwidth remaining percent of 2 on one class, and another queue with a bandwidth remaining percent of 4 on another class. The bandwidth remaining percent of 4 will be scheduled twice as often as the bandwidth remaining percent of 2.

The total bandwidth percent allocation for the policy can exceed 100. For example, you can configure a queue with a bandwidth remaining percent of 50, and another queue with a bandwidth remaining precent of 100.

Weighted Tail Drop

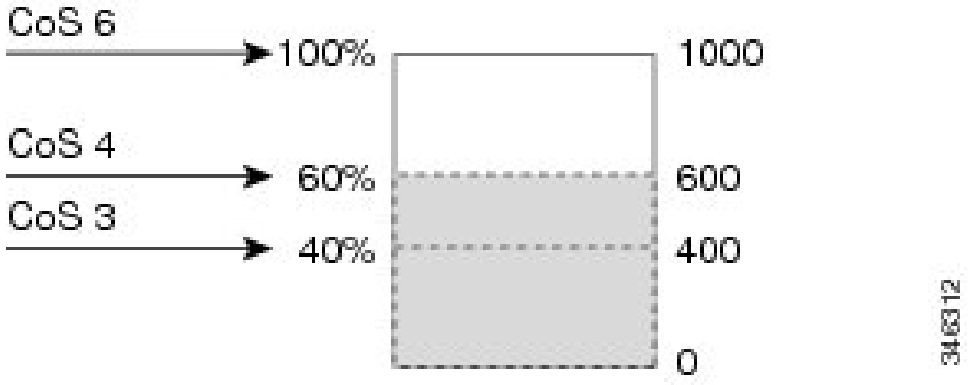
The device egress queues use an enhanced version of the tail-drop congestion-avoidance mechanism called weighted tail drop (WTD). WTD is implemented on queues to manage the queue lengths and to provide drop precedences for different traffic classifications.

As a frame is enqueued to a particular queue, WTD uses the frame's assigned QoS label to subject it to different thresholds. If the threshold is exceeded for that QoS label (the space available in the destination queue is less than the size of the frame), the device drops the frame.

Each queue has three configurable threshold values. The QoS label determines which of the three threshold values is subjected to the frame.

Figure 140: WTD and Queue Operation

The following figure shows an example of WTD operating on a queue whose size is 1000 frames. Three drop percentages are configured: 40 percent (400 frames), 60 percent (600 frames), and 100 percent (1000 frames). These percentages indicate that up to 400 frames can be queued at the 40-percent threshold, up to 600 frames at the 60-percent threshold, and up to 1000 frames at the 100-percent threshold.



In the example, CoS value 6 has a greater importance than the other CoS values, and is assigned to the 100-percent drop threshold (queue-full state). CoS values 4 is assigned to the 60-percent threshold, and CoS values 3 is assigned to the 40-percent threshold. All of these threshold values are assigned using the **queue-limit cos** command.

Assuming the queue is already filled with 600 frames, and a new frame arrives. It contains CoS value 4 and is subjected to the 60-percent threshold. If this frame is added to the queue, the threshold will be exceeded, so the device drops it.

Weighted Tail Drop Default Values

The following are the Weighted Tail Drop (WTD) default values and the rules for configuring WTD threshold values.

- If you configure less than three queue-limit percentages for WTD, then WTD default values are assigned to these thresholds.

The following are the WTD threshold default values:

Table 132: WTD Threshold Default Values

Threshold	Default Value Percentage
0	80
1	90
2	400

- If 3 different WTD thresholds are configured, then the queues are programmed as configured.
- If 2 WTD thresholds are configured, then the maximum value percentage will be 400.
- If a WTD single threshold is configured as x, then the maximum value percentage will be 400.
  - If the value of x is less than 90, then threshold1=90 and threshold 0= x.
  - If the value of x equals 90, then threshold1=90, threshold 0=80.
  - If the value x is greater than 90, then threshold1=x, threshold 0=80.

Priority Queues

Each port supports eight egress queues, of which two can be given a priority.

You use the **priority level** policy class-map command to configure the priority for two classes. One of the classes has to be configured with a priority queue level 1, and the other class has to be configured with a priority queue level 2. Packets on these two queues are subjected to less latency with respect to other queues.

You cannot send 100 percent line rate traffic when a priority queue is configured. There can only be 99.6 percent line rate traffic with priority queue configured, ensuring a latency of less than 20 microseconds.

Priority Queue Policer

The switch supports configuration of policing rate on priority queue. Priority queue policer supports only a single-rate two-color policing.

Note Policing with table-maps is not supported.

Examples of Configuring Priority Queue Policer

Example 1

```
Policy Map priority-1
    Class prio1
      priority level 1
     police rate percent 10
       conform-action transmit
       exceed-action drop
    Class prio2
```

```
  priority level 2
 police rate percent 5
   conform-action transmit
   exceed-action drop
Class new
  bandwidth 20 (%)
```

Example 2

```
Policy Map priority-1
    Class prio1
      priority level 1 20 (%)
     police rate percent 10
       conform-action transmit
       exceed-action drop
    Class prio2
      priority level 2 25 (%)
     police rate percent 5
       conform-action transmit
       exceed-action drop
    Class new
      bandwidth 20 (%)
```

Queue Buffer

At boot time, when there is no policy map enabled on the wired port, there are two queues created by default. Wired ports can have a maximum of 8 queues configured using MQC-based policies. The following table shows which packets go into which one of the queues:

Table 133: Queue Threshold Mapping Table for DSCP, Precedence, and CoS

DSCP, Precedence or CoS	Queue	Threshold
Control Packets	0	2
Rest of Packets	1	2

Note You can guarantee the availability of buffers, set drop thresholds, and configure the maximum memory allocation for a queue. You use the **queue-buffers** policy-map class command to configure the queue buffers. You use the **queue-limit** policy-map class command to configure the maximum thresholds.

There are two types of buffer allocations: hard buffers, which are explicitly reserved for the queue, and soft buffers, which are available for other ports when unused by a given port.

For the wired port default, Queue 0 will be given 40 percent of the buffers that are available for the interfaces as hard buffers, that is 81 buffers are allocated for Queue 0 in the context of 1-gigabit ports, and 408 buffers in the context of 10-gigabit ports. The soft maximum for this queue is set to four times the hard buffer, which is 324 for 1-gigabit ports and 1632 for 10-gigabit ports, where 400 is the default maximum threshold that is configured for any queue.

Queue 1 does not have any hard buffers allocated. Soft buffers have a minimum allocation of 122 buffers for the 1-gigabit ports, 612 buffers for 10-gigabit ports, and 2448 buffers for 25-gigabit ports. The soft maximum allocation for Queue 1 is four times the soft minimum with 488 buffers for 1-gigabit ports, 2448 buffers for 10-gigabit ports, and 9792 buffers for 25-gigabit ports.

Note By default, Queue 0 is not a priority queue. A policy-map can enable Queue 0 to be a priority queue by using the **priority level** command. If Queue 0 is assigned a priority level of 1, the soft maximum limit for this queue is automatically set to the same value as the hard maximum limit.

Queue Buffer Allocation

The buffer allocation to any queue can be tuned using the **queue-buffers ratio** policy-map class configuration command.

Dynamic Threshold and Scaling

Traditionally, reserved buffers are statically allocated for each queue. No matter whether the queue is active or not, its buffers are held up by the queue. In addition, as the number of queues increases, the portion of the reserved buffers allocated for each queue can become smaller and smaller. Eventually, a situation may occur where there are not enough reserved buffers to support a jumbo frame for all queues.

The device supports Dynamic Thresholding and Scaling (DTS), which is a feature that provides a fair and efficient allocation of buffer resources. When congestion occurs, this DTS mechanism provides an elastic buffer allocation for the incoming data based on the occupancy of the global/port resources. Conceptually, DTS scales down the queue buffer allocation gradually as the resources are used up to leave room for other queues, and vice versa. This flexible method allows the buffers to be more efficiently and fairly utilized.

As mentioned in the previous sections, there are two limits configured on a queue—a hard limit and a soft limit.

Hard limit are not part of DTS. These buffers are available only for that queue. The sum of the hard limits should be less than the globally set up hard maximum limit. The global hard limit configured for egress queuing is currently set to 5239. In the default scenario when there are no MQC polices configured, the 24 1-gigabit ports would take up 24 * 81=1944, and the 4 10-gigabit ports would take up 4 * 408=1632, for a total of 3576 buffers, allowing room for more hard buffers to be allocated based upon the configuration.

Soft limit buffers participate in the DTS process. Additionally, some of the soft buffer allocations can exceed the global soft limit allocation. The global soft limit allocation for egress queuing is currently set to 8073. The sum of the hard and soft limits add up to 13312, which in turn translates to 3.4 MB. Because the sum of the soft buffer allocations can exceed the global limit, it allows a specific queue to use a large number of buffers when the system is lightly loaded. The DTS process dynamically adjusts the per-queue allocation as the system becomes more heavily loaded.

Weighted Random Early Detection

Weighted random early detection (WRED) is a mechanism to avoid congestion in networks. WRED reduces the chances of tail drop by selectively dropping packets when the output interface begins to show signs of congestion, thus avoiding large number of packet drops at once.

For more information about WRED, see Configuring Weighted Random Early Detection, on page 1957.

Trust Behavior

The following sections provide information about trust behavior.

Port Security on a Trusted Boundary for Cisco IP Phones

In a typical network, you connect a Cisco IP Phone to a device port and cascade devices that generate data packets from the back of the telephone. The Cisco IP Phone guarantees the voice quality through a shared data link by marking the CoS level of the voice packets as high priority (CoS = 5) and by marking the data packets as low priority (CoS = 0). Traffic sent from the telephone to the device is typically marked with a tag that uses the 802.1Q header. The header contains the VLAN information and the class of service (CoS) 3-bit field, which is the priority of the packet.

For most Cisco IP Phone configurations, the traffic sent from the telephone to the device should be trusted to ensure that voice traffic is properly prioritized over other types of traffic in the network. By using the **trust device** interface configuration command, you configure the device port to which the telephone is connected to trust the traffic received on that port.

With the trusted setting, you also can use the trusted boundary feature to prevent misuse of a high-priority queue if a user bypasses the telephone and connects the PC directly to the device. Without trusted boundary, the CoS labels generated by the PC are trusted by the device (because of the trusted CoS setting). By contrast, trusted boundary uses CDP to detect the presence of a Cisco IP Phone (such as the Cisco IP Phone 7910, 7935, 7940, and 7960) on a device port. If the telephone is not detected, the trusted boundary feature disables the trusted setting on the device port and prevents misuse of a high-priority queue. Note that the trusted boundary feature is not effective if the PC and Cisco IP Phone are connected to a hub that is connected to the device.

Trust Behavior for Wired Ports

In scenarios where the incoming packet type differs from the outgoing packet type, the trust behavior and the queuing behavior are explained in the following table. Note that the default trust mode for a port is DSCP based. The trust mode ‘falls back’ to CoS if the incoming packet is a pure Layer 2 packet. You can also change the trust setting from DSCP to CoS. This setting change is accomplished by using an MQC policy that has a class default with a 'set cos cos table default default-cos' action, where default-cos is the name of the table map created (which only performs a default copy).

For wired ports that are connected to the device (end points such as IP phones, laptops, cameras, telepresence units, or other devices), the trust device configuration is enabled on the interface. Their DSCP, precedence, or CoS values coming in from these end points are trusted by the device and therefore are retained in the absence of any explicit policy configuration.

The packets are enqueued to the appropriate queue per the default initial configuration. No priority queuing at the device is done by default. This is true for unicast and multicast packets.

Table 134: Trust and Queuing Behavior

Incoming Packet	Outgoing Packet	Trust Behavior	Queuing Behavior
Layer 3	Layer 3	Preserve DSCP/Precedence	Based on DSCP
Layer 2	Layer 2	Not applicable	Based on CoS
Tagged	Tagged	Preserve DSCP and CoS	Based on DSCP (trust DSCP takes precedence)
Layer 3	Tagged	Preserve DSCP, CoS is set to 0	Based on DSCP

Default Wired QoS Configuration

There are two queues configured by default on each wired interface on the device. All control traffic traverses and is processed through queue 0. All other traffic traverses and is processed through queue 1.

DSCP Maps

This section provides information about DSCP maps.

Default CoS-to-DSCP Map

When DSCP transparency mode is disabled, the DSCP values are derived from CoS as per the following table. If these values are not appropriate for your network, you need to modify them.

Table 135: Default CoS-to-DSCP Map

CoS Value	DSCP Value
0	0
1	8
2	16
3	24
4	32
5	40
6	48
7	56

Default IP-Precedence-to-DSCP Map

You use the IP-precedence-to-DSCP map to map IP precedence values in incoming packets to a DSCP value that QoS uses internally to represent the priority of the traffic. The following table shows the default IP-precedence-to-DSCP map. If these values are not appropriate for your network, you need to modify them.

Table 136: Default IP-Precedence-to-DSCP Map

IP Precedence Value	DSCP Value
0	0
1	8
2	16
3	24
4	32
5	40

IP Precedence Value	DSCP Value
6	48
7	56

Default DSCP-to-CoS Map

You use the DSCP-to-CoS map to generate a CoS value, which is used to select one of the four egress queues. The following table shows the default DSCP-to-CoS map. If these values are not appropriate for your network, you need to modify them.

Table 137: Default DSCP-to-CoS Map

DSCP Value	CoS Value
0–7	0
8–15	1
16–23	2
24–31	3
32–39	4
40–47	5
48–55	6
56–63	7

How to Configure QoS

How to Configure Class, Policy, and Maps

The following sections provide configuration information about class, policy, and maps.

Creating a Traffic Class

To create a traffic class containing match criteria, use the **class-map** command to specify the traffic class name, and then use the following **match** commands in class-map configuration mode, as needed.

Before you begin

All match commands specified in this configuration task are considered optional, but you must configure at least one match criterion for a class.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**class-map** *class-map name* { **match-any** } **Example:** `Device(config)# class-map test_1000` `Device(config-cmap)#`	Enters class map configuration mode. • Creates a class map to be used for matching packets to the class whose name you specify. • **match-any:** Any one of the match criteria must be met for traffic entering the traffic class to be classified as part of it.
Step 3	**match access-group** { *index number* | *name*} **Example:** `Device(config-cmap)# match access-group 100` `Device(config-cmap)#`	The following parameters are available for this command: • access-group • cos • dscp • group-object • ip • precedence • protocol • qos-group • vlan (Optional) For this example, enter the access-group ID: • Access list index (value from 1 to 2799) • Named access list
Step 4	**match cos** *cos value* **Example:** `Device(config-cmap)# match cos 2 3 4 5` `Device(config-cmap)#`	(Optional) Matches IEEE 802.1Q or ISL class of service (user) priority values. • Enters up to 4 CoS values separated by spaces (0 to 7).

	Command or Action	Purpose
Step 5	**match dscp** *dscp value* **Example:** `Device(config-cmap)# match dscp af11 af12` `Device(config-cmap)#`	(Optional) Matches the DSCP values in IPv4 and IPv6 packets.
Step 6	**match ip** { **dscp** *dscp value* | **precedence** *precedence value* } **Example:** `Device(config-cmap)# match ip dscp af11 af12` `Device(config-cmap)#`	(Optional) Matches IP values including the following: • **dscp**: Matches IP DSCP (DiffServ codepoints). • **precedence**: Matches IP precedence (0 to 7).
Step 7	**match qos-group** *qos group value* **Example:** `Device(config-cmap)# match qos-group 10` `Device(config-cmap)#`	(Optional) Matches QoS group value (from 0 to 31).
Step 8	**match vlan** *vlan value* **Example:** `Device(config-cmap)# match vlan 210` `Device(config-cmap)#`	(Optional) Matches a VLAN ID (from 1 to 4095).
Step 9	**end** **Example:** `Device(config-cmap)# end`	Saves the configuration changes.

What to do next

Configure the policy map.

Creating a Traffic Policy

To create a traffic policy, use the **policy-map** global configuration command to specify the traffic policy name.

The traffic class is associated with the traffic policy when the **class** command is used. The **class** command must be entered after you enter the policy map configuration mode. After entering the **class** command, the device is automatically in policy map class configuration mode, which is where the QoS policies for the traffic policy are defined.

The following policy map class-actions are supported:

- bandwidth: Bandwidth configuration options.
- exit: Exits from the QoS class action configuration mode.
- no: Negates or sets default values for the command.
- police: Policer configuration options.
- priority: Strict scheduling priority configuration options for this class.
- queue-buffers: Queue buffer configuration options.
- queue-limit: Queue maximum threshold for Weighted Tail Drop (WTD) configuration options.
- service-policy: Configures the QoS service policy.
- set: Sets QoS values using the following options:
  - CoS values
  - DSCP values
  - Precedence values
  - QoS group values
- shape: Traffic-shaping configuration options.

Before you begin

You should have first created a class map.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**policy-map** *policy-map name* **Example:** `Device(config)# policy-map test_2000` `Device(config-pmap)#`	Enters policy map configuration mode. Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** {*class-name* | **class-default**} **Example:** `Device(config-pmap)# class test_1000` `Device(config-pmap-c)#`	Specifies the name of the class whose policy you want to create or change. You can also create a system default class for unclassified packets.

	Command or Action	Purpose
Step 4	**bandwidth** { *k* | **percent** *percentage* | **remaining** {*percent* | *ratio*} } **Example:** `Device(config-pmap-c)# bandwidth 500` `Device(config-pmap-c)#`	(Optional) Sets the bandwidth using one of the following: • *Kb/s*: Kilobits per second, enter a value between 100 and 100000000 for Kb/s. • **percent**: Enter the percentage of the total bandwidth to be used for this policy map. • **remaining**: Enter the percentage ratio of the remaining bandwidth. For a more detailed example of this command and its usage, see Configuring Bandwidth, on page 1928.
Step 5	**exit** **Example:** `Device(config-pmap-c)# exit` `Device(config-pmap-c)#`	(Optional) Exits from QoS class action configuration mode.
Step 6	**no** **Example:** `Device(config-pmap-c)# no` `Device(config-pmap-c)#`	(Optional) Negates the command.
Step 7	**police** {*target_bit_rate* | **cir** | **rate**} **Example:** `Device(config-pmap-c)# police 100000` `Device(config-pmap-c)#`	(Optional) Configures the policer: • *target_bit_rate*: Enter the bit rate per second, enter a value between 8000 and 10000000000. • **cir**: Committed Information Rate • **rate**: Specify police rate, PCR for hierarchical policies or SCR for single-level ATM 4.0 policer policies. For a more detailed example of this command and its usage, see Configuring Police , on page 1930.
Step 8	**Example:** `Device(config-pmap-c)#` `Device(config-pmap-c)#`	(Optional) Sets the strict scheduling priority for this class. Command options include: • **level**: Establishes a multi-level priority queue. Enter a value (1 or 2).

	Command or Action	Purpose
		For a more detailed example of this command and its usage, see Configuring Priority, on page 1932.
Step 9	**queue-buffers ratio** *ratio limit* **Example:** `Device(config-pmap-c)# queue-buffers ratio 10` `Device(config-pmap-c)#`	(Optional) Configures the queue buffer for the class. Enter the queue buffers ratio limit (0 to 100). For a more detailed example of this command and its usage, see Configuring Queue Buffers , on page 1936.
Step 10	**queue-limit** {*packets* | **cos** | **dscp** | **percent**} **Example:** `Device(config-pmap-c)# queue-limit cos 7 percent 50` `Device(config-pmap-c)#`	(Optional) Specifies the queue maximum threshold for the tail drop: • *packets*: Packets by default, enter a value between 1 to 2000000. • **cos**: Enter the parameters for each COS value. • **dscp**: Enter the parameters for each DSCP value. • **percent**: Enter the percentage for the threshold. For a more detailed example of this command and its usage, see Configuring Queue Limits, on page 1938.
Step 11	**service-policy** *policy-map name* **Example:** `Device(config-pmap-c)# service-policy test_2000` `Device(config-pmap-c)#`	(Optional) Configures the QoS service policy.
Step 12	**set** {**cos** | **dscp** | **ip** | **precedence** | **qos-group** | **wlan**} **Example:** `Device(config-pmap-c)# set cos 7` `Device(config-pmap-c)#`	(Optional) Sets the QoS values. Possible QoS configuration values include: • **cos**: Sets the IEEE 802.1Q/ISL class of service/user priority. • **dscp**: Sets DSCP in IP(v4) and IPv6 packets. • **ip**: Sets IP specific values. • **precedence**: Sets precedence in IP(v4) and IPv6 packet. • **qos-group**: Sets the QoS Group.

	Command or Action	Purpose
Step 13	**shape average** {*target _bit_rate* | **percent**} **Example:** `Device(config-pmap-c) #shape average percent 50` `Device(config-pmap-c) #`	(Optional) Sets the traffic shaping. Command parameters include: • *target_bit_rate*: Target bit rate. • **percent**: Percentage of interface bandwidth for Committed Information Rate. For a more detailed example of this command and its usage, see Configuring Shaping, on page 1941.
Step 14	**end** **Example:** `Device(config-pmap-c) #end` `Device(config-pmap-c) #`	Saves the configuration changes.

What to do next

Configure the interface.

Configuring Class-Based Packet Marking

This procedure explains how to configure the following class-based packet marking features on your device:

- CoS value
- DSCP value
- IP value
- Precedence value
- QoS group value

Before you begin

You should have created a class map and a policy map before beginning this procedure.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**policy-map** *policy name* **Example:** `Device(config)# policy-map policy1` `Device(config-pmap)#`	Enters policy map configuration mode. Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class name* **Example:** `Device(config-pmap)# class class1` `Device(config-pmap-c)#`	Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change. Command options for policy class map configuration mode include the following: • **bandwidth**: Bandwidth configuration options. • **exit**: Exits from the QoS class action configuration mode. • **no**: Negates or sets default values for the command. • **police**: Policer configuration options. • **priority**: Strict scheduling priority configuration options for this class. • **queue-buffers**: Queue buffer configuration options. • **queue-limit**: Queue maximum threshold for Weighted Tail Drop (WTD) configuration options. • **service-policy**: Configures the QoS service policy. • **set**: Sets QoS values using the following options: • CoS values • DSCP values • Precedence values • QoS group values • **shape**: Traffic-shaping configuration options. **Note** This procedure describes the available configurations using **set** command options.

	Command or Action	Purpose
		The other command options (**bandwidth**) are described in other sections of this guide. Although this task lists all of the possible **set** commands, only one **set** command is supported per class.
Step 4	**set cos** {*cos value* | **cos table** *table-map name* | **dscp table** *table-map name* | **precedence table** *table-map name* | **qos-group table** *table-map name* | } **Example:** `Device(config-pmap)# set cos 5` `Device(config-pmap)#`	(Optional) Sets the specific IEEE 802.1Q Layer 2 CoS value of an outgoing packet. Values are from 0 to7. You can also set the following values using the **set cos** command: • **cos table**: Sets the CoS value based on a table map. • **dscp table**: Sets the code point value based on a table map. • **precedence table**: Sets the code point value based on a table map. • **qos-group table**: Sets the CoS value from QoS group based on a table map.
Step 5	**set dscp** {*dscp value* | **default** | **dscp table** *table-map name* | **ef** | **precedence table** *table-map name* | **qos-group table** *table-map name* | *table-map name*} **Example:** `Device(config-pmap)# set dscp af11` `Device(config-pmap)#`	(Optional) Sets the DSCP value. In addition to setting specific DSCP values, you can also set the following using the **set dscp** command: • **default**: Matches packets with default DSCP value (000000). • **dscp table**: Sets the packet DSCP value from DSCP based on a table map. • **ef**: Matches packets with EF DSCP value (101110). • **precedence table**: Sets the packet DSCP value from precedence based on a table map. • **qos-group table**: Sets the packet DSCP value from a QoS group based upon a table map.
Step 6	**set ip** {**dscp** | **precedence**} **Example:** `Device(config-pmap)# set ip dscp c3`	(Optional) Sets IP specific values. These values are either IP DSCP or IP precedence values. You can set the following values using the **set ip dscp** command:

	Command or Action	Purpose
	`Device(config-pmap)#`	• *dscp value*: Sets a specific DSCP value. • **default**: Matches packets with default DSCP value (000000). • **dscp table**: Sets the packet DSCP value from DSCP based on a table map. • **ef**: Matches packets with EF DSCP value (101110). • **precedence table**: Sets the packet DSCP value from precedence based on a table map. • **qos-group table**: Sets the packet DSCP value from a QoS group based upon a table map. You can set the following values using the **set ip precedence** command: • *precedence value*: Sets the precedence value (from 0 to 7) . • **cos table**: Sets the packet precedence value from Layer 2 CoS based on a table map. • **dscp table**: Sets the packet precedence from DSCP value based on a table map. • **precedence table**: Sets the precedence value from precedence based on a table map • **qos-group table**: Sets the precedence value from a QoS group based upon a table map.
Step 7	**set precedence** {*precedence value* | **cos table** *table-map name* | **dscp table** *table-map name* | **precedence table** *table-map name* | **qos-group table** *table-map name*} **Example:** `Device(config-pmap)# set precedence 5` `Device(config-pmap)#`	(Optional) Sets precedence values in IPv4 and IPv6 packets. You can set the following values using the **set precedence** command: • *precedence value*: Sets the precedence value (from 0 to 7) . • **cos table**: Sets the packet precedence value from Layer 2 CoS on a table map. • **dscp table**: Sets the packet precedence from DSCP value on a table map.

	Command or Action	Purpose
		• **precedence table**: Sets the precedence value from precedence based on a table map. • **qos-group table**: Sets the precedence value from a QoS group based upon a table map.
Step 8	**set qos-group** {*qos-group value* | **dscp table** *table-map name* | **precedence table** *table-map name*} **Example:** `Device(config-pmap)# set qos-group 10` `Device(config-pmap)#`	(Optional) Sets QoS group values. You can set the following values using this command: • *qos-group value*: A number from 1 to 31. • **dscp table**: Sets the code point value from DSCP based on a table map. • **precedence table**: Sets the code point value from precedence based on a table map.
Step 9	**end** **Example:** `Device(config-pmap)# end` `Device#`	Saves configuration changes.
Step 10	**show policy-map** **Example:** `Device# show policy-map`	(Optional) Displays policy configuration information for all classes configured for all service policies.

What to do next

Attach the traffic policy to an interface using the **service-policy** command.

Attaching a Traffic Policy to an Interface

After the traffic class and traffic policy are created, you must use the **service-policy** interface configuration command to attach a traffic policy to an interface, and to specify the direction in which the policy should be applied (either on packets coming into the interface or packets leaving the interface).

Before you begin

A traffic class and traffic policy must be created before attaching a traffic policy to an interface.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type* **Example:**	
Step 3	**service-policy** {**input** *policy-map* | **output** *policy-map*} **Example:** `Device(config-if)# service-policy output policy_map_01` `Device(config-if)#`	Attaches a policy map to an input or output interface. This policy map is then used as the service policy for that interface. In this example, the traffic policy evaluates all traffic leaving that interface.
Step 4	**end** **Example:** `Device(config-if)# end` `Device#`	Saves configuration changes.
Step 5	**show policy map** **Example:** `Device# show policy map`	(Optional) Displays statistics for the policy on the specified interface.

What to do next

Proceed to attach any other traffic policy to an interface, and to specify the direction in which the policy should be applied.

Classifying, Policing, and Marking Traffic on Physical Ports by Using Policy Maps

You can configure a nonhierarchical policy map on a physical port that specifies which traffic class to act on. Actions supported are remarking and policing.

Before you begin

You should have already decided upon the classification, policing, and marking of your network traffic by policy maps prior to beginning this procedure.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**class-map** {*class-map name* | **match-any**} **Example:** `Device(config)# class-map ipclass1` `Device(config-cmap)# exit` `Device(config)#`	Enters class map configuration mode. • Creates a class map to be used for matching packets to the class whose name you specify. • If you specify **match-any**, one of the match criteria must be met for traffic entering the traffic class to be classified as part of the traffic class. This is the default.
Step 3	**match access-group** { *access list index* | *access list name* } **Example:** `Device(config-cmap)# match access-group 1000` `Device(config-cmap)# exit` `Device(config)#`	The following parameters are available for this command: • access-group • cos • dscp • group-object • ip • precedence • protocol • qos-group • vlan (Optional) For this example, enter the access-group ID: • Access list index (value from 1 to 2799) • Named access list
Step 4	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map ipclass1`	Creates a policy map by entering the policy map name, and enters policy-map configuration mode. By default, no policy maps are defined.

	Command or Action	Purpose
	`Device(config-pmap)#`	
Step 5	**class** {*class-map-name* | **class-default**} **Example:** `Device(config-pmap)# class ipclass1` `Device(config-pmap-c)#`	Defines a traffic classification, and enter policy-map class configuration mode. By default, no policy map class-maps are defined. If a traffic class has already been defined by using the **class-map** global configuration command, specify its name for *class-map-name* in this command. A **class-default** traffic class is predefined and can be added to any policy. It is always placed at the end of a policy map. With an implied **match any** included in the **class-default** class, all packets that have not already matched the other traffic classes will match **class-default**.
Step 6	**set** {**cos** | **dscp** | **ip** | **precedence** | **qos-group** | } **Example:** `Device(config-pmap-c)# set dscp 45` `Device(config-pmap-c)#`	(Optional) Sets the QoS values. Possible QoS configuration values include: • **cos**: Sets the IEEE 802.1Q/ISL class of service/user priority. • **dscp**: Sets DSCP in IP(v4) and IPv6 packets. • **ip**: Sets IP specific values. • **precedence**: Sets precedence in IP(v4) and IPv6 packet. • **qos-group**: Sets QoS group. In this example, the **set dscp** command classifies the IP traffic by setting a new DSCP value in the packet.
Step 7	**police** {*target_bit_rate* | **cir** | **rate** } **Example:** `Device(config-pmap-c)# police 100000 conform-action transmit exceed-action drop` `Device(config-pmap-c)#`	(Optional) Configures the policer: • *target_bit_rate*: Specifies the bit rate per second, enter a value between 8000 and 10000000000. • **cir**: Committed Information Rate. • **rate**: Specifies the police rate PCR for hierarchical policies. In this example, the **police** command adds a policer to the class where any traffic beyond the 100000 set target bit rate is dropped.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Device(config-pmap-c)# exit`	Returns to policy map configuration mode.
Step 9	**exit** **Example:** `Device(config-pmap)# exit`	Returns to global configuration mode.
Step 10	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to attach to the policy map, and enters interface configuration mode. Valid interfaces include physical ports.
Step 11	**service-policy input** *policy-map-name* **Example:** `Device(config-if)# service-policy input flowit`	Specifies the policy-map name, and applies it to an ingress port. Only one policy map per ingress port is supported.
Step 12	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 13	**show policy-map** [*policy-map-name* [**class** *class-map-name*]] **Example:** `Device# show policy-map ipclass1`	(Optional) Verifies your entries.
Step 14	**copy running-config startup-config** **Example:** `Device# copy-running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

If applicable to your QoS configuration, configure classification, policing, and marking of traffic on SVIs by using policy maps.

Classifying and Marking Traffic by Using Policy Maps

Before you begin

You should have already decided upon the classification, policing, and marking of your network traffic by using policy maps prior to beginning this procedure.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**class-map** {*class-map name* | **match-any**} **Example:** `Device(config)# class-map class_vlan100`	Enters class map configuration mode. • Creates a class map to be used for matching packets to the class whose name you specify. • If you specify **match-any**, one of the match criteria must be met for traffic entering the traffic class to be classified as part of the traffic class.
Step 3	**match vlan** *vlan number* **Example:** `Device(config-cmap)# match vlan 100` `Device(config-cmap)# exit` `Device(config)#`	Specifies the VLAN to match to the class map.
Step 4	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map policy_vlan100` `Device(config-pmap)#`	Creates a policy map by entering the policy map name, and enters policy-map configuration mode. By default, no policy maps are defined.
Step 5	**description** *description* **Example:**	(Optional) Enters a description of the policy map.

	Command or Action	Purpose
	`Device(config-pmap)# description vlan 100`	
Step 6	**class** {*class-map-name* | **class-default**} **Example:** `Device(config-pmap)# class class_vlan100` `Device(config-pmap-c)#`	Defines a traffic classification, and enters the policy-map class configuration mode. By default, no policy map class-maps are defined. If a traffic class has already been defined by using the **class-map** global configuration command, specify its name for *class-map-name* in this command. A **class-default** traffic class is predefined and can be added to any policy. It is always placed at the end of a policy map. With an implied **match any** included in the **class-default** class, all packets that have not already matched the other traffic classes will match **class-default**.
Step 7	**set** {**cos** | **dscp** | **ip** | **precedence** | **qos-group** | } **Example:** `Device(config-pmap-c)# set dscp af23` `Device(config-pmap-c)#`	(Optional) Sets the QoS values. Possible QoS configuration values include: • **cos**: Sets the IEEE 802.1Q/ISL class of service/user priority. • **dscp**: Sets DSCP in IP(v4) and IPv6 packets. • **ip**: Sets IP specific values. • **precedence**: Sets precedence in IP(v4) and IPv6 packet. • **qos-group**: Sets QoS group. In this example, the **set dscp** command classifies the IP traffic by matching the packets with a DSCP value of AF23 (010010).
Step 8	**exit** **Example:** `Device(config-pmap-c)# exit`	Returns to policy map configuration mode.
Step 9	**exit** **Example:** `Device(config-pmap)# exit`	Returns to global configuration mode.

	Command or Action	Purpose
Step 10	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the port to attach to the policy map, and enters interface configuration mode. Valid interfaces include physical ports.
Step 11	**service-policy input** *policy-map-name* **Example:** `Device(config-if)# service-policy input policy_vlan100`	Specifies the policy-map name, and applies it to an ingress port. Only one policy map per ingress port is supported.
Step 12	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 13	**show policy-map** [*policy-map-name* [**class** *class-map-name*]] **Example:** `Device# show policy-map`	(Optional) Verifies your entries.
Step 14	**copy running-config startup-config** **Example:** `Device# copy-running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Table Maps

Table maps are a form of marking, and also enable the mapping and conversion of one field to another using a table. For example, a table map can be used to map and convert a Layer 2 CoS setting to a precedence value in Layer 3.

Note

- A table map can be referenced in multiple policies or multiple times in the same policy.
- A table map configured for a custom output policy under the default class-map, takes affect for all DSCP traffic regardless of which class map the traffic is classified for. The workaround is to remove the table map and configure the **set dscp** command under the default class to change the DSCP marking for classified traffic. If there is any non-queuing action (policer or marking) on a user-defined class, then the packet retains its value or remarks in the user-defined class itself.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**table-map** *name* {**default** {*default value* | **copy** | **ignore**} | **exit** | **map** {**from** *from value* **to** *to value* } | **no**} **Example:** `Device(config)# table-map table01` `Device(config-tablemap)#`	Creates a table map and enters the table map configuration mode. In table map configuration mode, you can perform the following tasks: • **default**: Configures the table map default value, or sets the default behavior for a value not found in the table map to copy or ignore. • **exit**: Exits from the table map configuration mode. • **map**: Maps a *from* to a *to* value in the table map. • **no**: Negates or sets the default values of the command.
Step 3	**map from** *value* **to** *value* **Example:** `Device(config-tablemap)# map from 0 to 2` `Device(config-tablemap)# map from 1 to 4` `Device(config-tablemap)# map from 24 to 3` `Device(config-tablemap)# map from 40 to 6` `Device(config-tablemap)# default 0` `Device(config-tablemap)#`	In this step, packets with DSCP values 0 are marked to the CoS value 2, DSCP value 1 to the CoS value 4, DSCP value 24 to the CoS value 3, DSCP value 40 to the CoS value 6 and all others to the CoS value 0. **Note** The mapping from CoS values to DSCP values in this example is configured by using the **set** policy map class configuration command as described in a later step in this procedure.
Step 4	**exit** **Example:** `Device(config-tablemap)# exit` `Device(config)#`	Returns to global configuration mode.
Step 5	**exit** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config) exit` `Device#`	
Step 6	**show table-map** **Example:** `Device# show table-map` `Table Map table01` `from 0 to 2` `from 1 to 4` `from 24 to 3` `from 40 to 6` `default 0`	Displays the table map configuration.
Step 7	**configure terminal** **Example:** `Device# configure terminal` `Device(config)#`	Enters global configuration mode.
Step 8	**policy-map** **Example:** `Device(config)# policy-map table-policy` `Device(config-pmap)#`	Configures the policy map for the table map.
Step 9	**class class-default** **Example:** `Device(config-pmap)# class class-default` `Device(config-pmap-c)#`	Matches the class to the system default.
Step 10	**set cos dscp table** *table map name* **Example:** `Device(config-pmap-c)# set cos dscp table table01` `Device(config-pmap-c)#`	If this policy is applied on input port, that port will have trust DSCP enabled on that port and marking will take place depending upon the specified table map.
Step 11	**end** **Example:** `Device(config-pmap-c)# end` `Device#`	Returns to privileged EXEC mode.

What to do next

Configure any additional policy maps for QoS for your network. After creating your policy maps, attach the traffic policy or polices to an interface using the **service-policy** command.

How to Configure QoS Features and Functionality

The following sections provide configurational information about QoS features and functionality.

Configuring Bandwidth

This procedure explains how to configure bandwidth on your device.

Before you begin

You should have created a class map for bandwidth before beginning this procedure.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**policy-map** *policy name* **Example:** `Device(config)# policy-map policy_bandwidth01` `Device(config-pmap)#`	Enters policy map configuration mode. Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class name* **Example:** `Device(config-pmap)# class class_bandwidth01` `Device(config-pmap-c)#`	Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change. Command options for policy class map configuration mode include the following: • *word*: Class map name. • **class-default**: System default class matching any otherwise unclassified packets.
Step 4	**bandwidth** {*Kb/s* | **percent** *percentage* | **remaining** {**ratio** *ratio* }} **Example:** `Device(config-pmap-c)# bandwidth 200000`	Configures the bandwidth for the policy map. The parameters include: • *Kb/s*: Configures a specific value in kilobits per second (from 100 to 100000000).

	Command or Action	Purpose
	`Device(config-pmap-c)#`	• **percent**: Allocates minimum bandwidth to a particular class based on a percentage. The queue can oversubscribe bandwidth in case other queues do not utilize the entire port bandwidth. The total sum cannot exceed 100 percent, and in case it is less than 100 percent, the rest of the bandwidth is equally divided along all bandwidth queues. • **remaining**: Allocates minimum bandwidth to a particular class. The queue can oversubscribe bandwidth in case other queues do not utilize entire port bandwidth. The total sum cannot exceed 100 percent. It is preferred to use this command when the **priority** command is used for certain queues in the policy. You can also assign ratios rather than percentages to each queue; the queues will be assigned certain weights which are inline with these ratios. Ratios can range from 0 to 100. Total bandwidth ratio allocation for the policy in this case can exceed 100. **Note** You cannot mix bandwidth types on a policy map. For example, you cannot configure bandwidth in a single policy map using both a bandwidth percent and in kilobits per second.
Step 5	**end** **Example:** `Device(config-pmap-c)# end` `Device#`	Saves configuration changes.
Step 6	**show policy-map** **Example:** `Device# show policy-map`	(Optional) Displays policy configuration information for all classes configured for all service policies.

What to do next

Configure any additional policy maps for QoS for your network. After creating the policy maps, attach the traffic policy or polices to an interface using the **service-policy** command.

Configuring Police

This procedure explains how to configure policing on your device.

Before you begin

You should have created a class map for policing before beginning this procedure.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**policy-map** *policy name* **Example:** `Device(config)# policy-map policy_police01` `Device(config-pmap)#`	Enters policy map configuration mode. Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class name* **Example:** `Device(config-pmap)# class class_police01` `Device(config-pmap-c)#`	Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change. Command options for policy class map configuration mode include the following: • *word*: Class map name. • **class-default**: System default class matching any otherwise unclassified packets.
Step 4	**police** {*target_bit_rate* [*burst bytes* | **bc** | **conform-action** | **pir**] | **cir** {*target_bit_rate* | **percent** *percentage*} | **rate** {*target_bit_rate* | **percent** *percentage*} **conform-action transmit exceed-action** {**drop** [**violate action**] | **set-cos-transmit** | **set-dscp-transmit** | **set-prec-transmit** | **transmit** [**violate action**] }} **Example:** `Device(config-pmap-c)# police 8000 conform-action transmit exceed-action drop` `Device(config-pmap-c)#`	The following **police** subcommand options are available: • *target_bit_rate*: Bits per second (from 8000 to 10000000000). • *burst bytes*: Enter a value from 1000 to 512000000. • **bc**: Conform burst. • **conform-action**: Action taken when rate is less than conform burst. • **pir**: Peak Information Rate. • **cir**: Committed Information Rate. • *target_bit_rate*: Target bit rate (8000 to10000000000).

	Command or Action	Purpose
		• **percent**: Percentage of interface bandwidth for CIR. • **rate**: Specifies the police rate, PCR for hierarchical policies, or SCR for single-level ATM 4.0 policer policies. • *target_bit_rate*: Target Bit Rate (8000 to 10000000000). • **percent**: Percentage of interface bandwidth for rate. The following **police conform-action transmit exceed-action** subcommand options are available: • **drop**: Drops the packet. • **set-cos-transmit**: Sets the CoS value and sends it. • **set-dscp-transmit**: Sets the DSCP value and sends it. • **set-prec-transmit**: Rewrites the packet precedence and sends it. • **transmit**: Transmits the packet. **Note** Policer-based markdown actions are only supported using table maps. Only one markdown table map is allowed for each marking field in the device.
Step 5	**end** **Example:** `Device(config-pmap-c)# end` `Device#`	Saves configuration changes.
Step 6	**show policy-map** **Example:** `Device# show policy-map`	(Optional) Displays policy configuration information for all classes configured for all service policies. **Note** The **show policy-map** command output does not display counters for conformed bytes and exceeded bytes

What to do next

Configure any additional policy maps for QoS for your network. After creating your policy maps, attach the traffic policy or polices to an interface using the **service-policy** command.

Configuring Priority

This procedure explains how to configure priority on your device.

Note The device supports giving priority to specified queues. There are two priority levels available (1 and 2). Queues supporting voice and video should be assigned a priority level of 1.

Before you begin

You should have created a class map for priority before beginning this procedure.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**policy-map** *policy name* **Example:** `Device(config)# policy-map policy_priority01` `Device(config-pmap)#`	Enters policy map configuration mode. Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class name* **Example:** `Device(config-pmap)# class class_priority01` `Device(config-pmap-c)#`	Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change. Command options for policy class map configuration mode include the following: • *word*: Class map name. • **class-default**: System default class matching any otherwise unclassified packets.
Step 4	**Example:** `Device(config-pmap-c)# priority level 1` `Device(config-pmap-c)#`	(Optional) The **priority** command assigns a strict scheduling priority for the class. The command options include:

	Command or Action	Purpose
		• **level** *level_value*: Specifies the multilevel (1-2) priority queue. **Note** Priority level 1 is more important than priority level 2. Priority level 1 reserves bandwidth that is processed first for QoS, so its latency is very low. Both priority level 1 and 2 reserve bandwidth.
Step 5	**end** **Example:** `Device(config-pmap-c)# end` `Device#`	Saves configuration changes.
Step 6	**show policy-map** **Example:** `Device# show policy-map`	(Optional) Displays policy configuration information for all classes configured for all service policies.

What to do next

Configure any additional policy maps for QoS for your network. After creating your policy maps, attach the traffic policy or polices to an interface using the **service-policy** command.

Configuring SGT based QoS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**class-map** *class-map-name* {**match-any** | **match-all** } **Example:** `Device(config)# class-map c1`	Specifies the class-map and enters class-map configuration mode.
Step 3	**match security-group source tag** *sgt-number* **Example:**	Configures the value for security-group source security tag.

	Command or Action	Purpose
	`Device(config-cmap)# match security-group source tag 1000`	
Step 4	**match security-group destination tag** *dgt-number* **Example:** `Device(config-cmap)# match security-group destination tag 2000`	Configures the value for security-group destination security tag.
Step 5	**exit** **Example:** `Device(config-cmap)# exit` `Device#`	Exits route-map configuration mode and returns to global configuration mode.
Step 6	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map pin` `Device(config-pmap)#`	Specifies the policy-map and enters policy-map configuration mode. *policy-map-name* is the name of the child policy map. The name can be a maximum of 40 alphanumeric characters.
Step 7	**class** *class-name* **Example:** `Device(config-pmap)# class c1` `Device(config-pmap-c)#`	Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change. Command options for policy class map configuration mode include the following: • *word*: Class map name. • **class-default**: System default class matching any otherwise unclassified packets.
Step 8	**set dscp** *dscp-value* **Example:** `Device(config-pmap-c)# set dscp af11`	Configures the Differentiated Services CodePoint (DSCP) value.
Step 9	**end** **Example:** `Device(config-pmap-c)# end` `Device#`	Saves configuration changes. Exits class-map configuration mode and enters global configuration mode.
Step 10	**interface** *interface-num* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface and enters the interface configuration mode.

	Command or Action	Purpose
Step 11	**service-policy** { **input** | **output** } *policy-map-name* **Example:** `Device(config-if)#service-policy input pin`	Assigns policy-map to the ingress of the interface.
Step 12	**end** **Example:** `Device(config-if)# end` `Device#`	Saves configuration changes. Exits interface configuration mode and enters global configuration mode.

Configuration Example for SGT based QoS Classification

The following is an sample configuration for SGT based QoS on an interface:

```
ip access-list role-based sgt_acl
 10 permit ip
cts role-based sgt-map 24.0.0.0/8 sgt 24
cts role-based enforcement
cts role-based permissions from 24 to 24 sgt_acl

class-map match-all c1
 match protocol attribute business-relevance business-relevant
 match protocol attribute traffic-class ops-admin-mgmt
 match security-group destination tag 24
 match security-group source tag 24

policy-map pin
 class c1
  set dscp af11
 class class-default
  set dscp af12

interface gigabitethernet 1/1
 no switchport
 ip address 24.1.1.2 255.255.255.0
 service-policy input pin
 ip nbar protocol-discovery
```

Configuring Queues and Shaping

The following sections provide configurational information about queueing and shaping.

Configuring Egress Queue Characteristics

Depending on the complexity of your network and your QoS solution, you may need to perform all of the procedures in this section. You need to make decisions about these characteristics:

- Which packets are mapped by DSCP, CoS, or QoS group value to each queue and threshold ID?

- What drop percentage thresholds apply to the queues, and how much reserved and maximum memory is needed for the traffic type?
- How much of the fixed buffer space is allocated to the queues?
- Does the bandwidth of the port need to be rate limited?
- How often should the egress queues be serviced and which technique (shaped, shared, or both) should be used?

Note You can only configure the egress queues on the device.

Configuring Queue Buffers

The device allows you to allocate buffers to queues. If there is no allocation made to buffers, then they are divided equally for all queues. You can use the queue-buffer ratio to divide it in a particular ratio. Since by default DTS (Dynamic Threshold and Scaling) is active on all queues, these are soft buffers.

Note Queue-buffer ratio cannot be configured with a queue-limit.

Before you begin

The following are prerequisites for this procedure:

- You should have created a class map for the queue buffer before beginning this procedure.
- You must have configured either bandwidth, shape, or priority on the policy map prior to configuring the queue buffers.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**policy-map** *policy name* **Example:** `Device(config)# policy-map policy_queuebuffer01` `Device(config-pmap)#`	Enters policy map configuration mode. Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.

	Command or Action	Purpose
Step 3	**class** *class name* **Example:** `Device(config-pmap)# class class_queuebuffer01` `Device(config-pmap-c)#`	Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change. Command options for policy class map configuration mode include the following: • *word*: Class map name. • **class-default**: System default class matching any otherwise unclassified packets.
Step 4	**bandwidth** {*Kb/s* | **percent** *percentage* | **remaining** {**ratio** *ratio value* }} **Example:** `Device(config-pmap-c)# bandwidth percent 80` `Device(config-pmap-c)#`	Configures the bandwidth for the policy map. The command parameters include: • *Kb/s*: Use this command to configure a specific value. The range is 20000 to 100000000. • **percent**: Allocates a minimum bandwidth to a particular class using a percentage. The queue can oversubscribe bandwidth in case other queues do not utilize the entire port bandwidth. The total sum cannot exceed 100 percent, and in case it is less than 100 percent, the rest of the bandwidth is equally divided along all bandwidth queues. • **remaining**: Allocates a minimum bandwidth to a particular class. The queue can oversubscribe bandwidth in case other queues do not utilize entire port bandwidth. The total sum cannot exceed 100 percent. It is preferred to use this command when the **priority** command is used for certain queues in the policy. You can also assign ratios rather than a percentage to each queue; the queues will be assigned certain weights that are inline with these ratios. Ratios can range from 0 to 100. Total bandwidth ratio allocation for the policy in this case can exceed 100. **Note** You cannot mix bandwidth types on a policy map.
Step 5	**queue-buffers** {**ratio** *ratio value*} **Example:**	Configures the relative buffer size for the queue. **Note**

	Command or Action	Purpose
	`Device(config-pmap-c)# queue-buffers ratio 10` `Device(config-pmap-c)#`	The sum of all configured buffers in a policy must be less than or equal to 100 percent. Unallocated buffers are are evenly distributed to all the remaining queues. Ensure sufficient buffers are allocated to all queues including the priority queues. **Note** Protocol Data Units(PDUs) for network control protocols such as spanning-tree and LACP utilize the priority queue or queue 0 (when a priority queue is not configured). Ensure sufficient buffers are allocated to these queues for the protocols to function.
Step 6	**end** **Example:** `Device(config-pmap-c)# end` `Device#`	Saves configuration changes.
Step 7	**show policy-map** **Example:** `Device# show policy-map`	(Optional) Displays policy configuration information for all classes configured for all service policies.

What to do next

Configure any additional policy maps for QoS for your network. After creating your policy maps, attach the traffic policy or polices to an interface using the **service-policy** command.

Configuring Queue Limits

You use queue limits to configure Weighted Tail Drop (WTD). WTD ensures the configuration of more than one threshold per queue. Each class of service is dropped at a different threshold value to provide for QoS differentiation. With the device, each queue has 3 explicit programmable threshold classes—0, 1, 2. Therefore, the enqueue/drop decision of each packet per queue is determined by the packet's threshold class assignment, which is determined by the DSCP, CoS, or QoS group field of the frame header.

WTD also uses a soft limit, and therefore you are allowed to configure the queue limit to up to 400 percent (maximum four times the reserved buffer from common pool). This soft limit prevents overrunning the common pool without impacting other features.

Note You can only configure queue limits on the device egress queues on wired ports.

Before you begin

The following are prerequisites for this procedure:

- You should have created a class map for the queue limits before beginning this procedure.
- You must have configured either bandwidth, shape, or priority on the policy map prior to configuring the queue limits.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**policy-map** *policy name* **Example:** `Device(config)# policy-map policy_queuelimit01` `Device(config-pmap)#`	Enters policy map configuration mode. Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class name* **Example:** `Device(config-pmap)# class class_queuelimit01` `Device(config-pmap-c)#`	Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change. Command options for policy class map configuration mode include the following: • *word*: Class map name. • **class-default**: System default class matching any otherwise unclassified packets.
Step 4	**bandwidth** {*Kb/s* | **percent** *percentage* | **remaining** {**ratio** *ratio value* }} **Example:** `Device(config-pmap-c)# bandwidth 500000` `Device(config-pmap-c)#`	Configures the bandwidth for the policy map. The parameters include: • *Kb/s*: Use this command to configure a specific value. The range is 20000 to 100000000. • **percent**: Allocates a minimum bandwidth to a particular class. The queue can oversubscribe bandwidth in case other queues do not utilize the entire port bandwidth. The total sum cannot exceed 100 percent, and in case it is less than 100 percent, the rest of the bandwidth is

	Command or Action	Purpose
		equally divided along all bandwidth queues. • **remaining**: Allocates a minimum bandwidth to a particular class. The queue can oversubscribe bandwidth in case other queues do not utilize entire port bandwidth. The total sum cannot exceed 100 percent. It is preferred to use this command when the **priority** command is used for certain queues in the policy. You can also assign ratios rather than a percentage to each queue; the queues will be assigned certain weights that are inline with these ratios. Ratios can range from 0 to 100. Total bandwidth ratio allocation for the policy in this case can exceed 100. **Note** You cannot mix bandwidth types on a policy map.
Step 5	**queue-limit** {*packets* **packets** | **cos** {*cos value* { *maximum threshold value* | **percent** *percentage* } | **values** {*cos value* | **percent** *percentage* } } | **dscp** {*dscp value* {*maximum threshold value* | **percent** *percentage*} | *match packet* {*maximum threshold value* | **percent** *percentage*} | **default** {*maximum threshold value* | **percent** *percentage*} | **ef** {*maximum threshold value* | **percent** *percentage*} | **dscp values** *dscp value*} | **percent** *percentage* }} **Example:** `Device(config-pmap-c)# queue-limit dscp 3 percent 20` `Device(config-pmap-c)# queue-limit dscp 4 percent 30` `Device(config-pmap-c)# queue-limit dscp 5 percent 40`	Sets the queue limit threshold percentage values. With every queue, there are three thresholds (0,1,2), and there are default values for each of these thresholds. Use this command to change the default or any other queue limit threshold setting. For example, if DSCP 3, 4, and 5 packets are being sent into a specific queue in a configuration, then you can use this command to set the threshold percentages for these three DSCP values. For additional information about queue limit threshold values, see #unique_2676. **Note** The device does not support absolute queue-limit percentages. The device only supports DSCP or CoS queue-limit percentages.
Step 6	**end** **Example:** `Device(config-pmap-c)# end` `Device#`	Saves configuration changes.

	Command or Action	Purpose
Step 7	**show policy-map** **Example:** `Device# show policy-map`	(Optional) Displays policy configuration information for all classes configured for all service policies.

What to do next

Proceed to configure any additional policy maps for QoS for your network. After creating your policy maps, proceed to attach the traffic policy or polices to an interface using the **service-policy** command.

Configuring Shaping

You use the **shape** command to configure shaping (maximum bandwidth) for a particular class. The queue's bandwidth is restricted to this value even though the port has additional bandwidth left. You can configure shaping as an average percent, as well as a shape average value in bits per second.

Before you begin

You should have created a class map for shaping before beginning this procedure.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**policy-map** *policy name* **Example:** `Device(config)# policy-map policy_shaping01` `Device(config-pmap)#`	Enters policy map configuration mode. Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class name* **Example:** `Device(config-pmap)# class class_shaping01` `Device(config-pmap-c)#`	Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change. Command options for policy class map configuration mode include the following: • *word*: Class map name. • **class-default**: System default class matching any otherwise unclassified packets.

	Command or Action	Purpose
Step 4	**shape average** {*target bit rate* | **percent** *percentage*} **Example:** `Device(config-pmap-c)# shape average percent 50` `Device(config-pmap-c)#`	Configures the average shape rate. You can configure the average shape rate by target bit rates (bits per second) or by percentage of interface bandwidth for the Committed Information Rate (CIR).
Step 5	**end** **Example:** `Device(config-pmap-c)# end` `Device#`	Saves configuration changes.
Step 6	**show policy-map** **Example:** `Device# show policy-map`	(Optional) Displays policy configuration information for all classes configured for all service policies.

What to do next

Configure any additional policy maps for QoS for your network. After creating your policy maps, attach the traffic policy or polices to an interface using the **service-policy** command.

Configuring Sharped Profile Queuing

This procedure explains how to configure sharped profile queuing on your switch:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**policy-map** *policy name* **Example:** `Device(config)# policy-map policy_shaping01` `Device(config-pmap)#`	Enters policy map configuration mode. Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy. *policy-map-name* is the name of the child policy map. The name can be a maximum of 40 alphanumeric characters.

	Command or Action	Purpose
Step 3	**class** *class name* **Example:** `Device(config-pmap)# class class_shaping01` `Device(config-pmap-c)#`	Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change. Command options for policy class map configuration mode include the following: • *word*: Class map name. • **class-default**: System default class matching any otherwise unclassified packets.
Step 4	**bandwidth** {*Kb/s* | **percent** *percentage* | **remaining** {**ratio** *ratio value*}} **Example:** `Device(config-pmap-c)# bandwidth 200000` `Device(config-pmap-c)#`	Configures the bandwidth for the policy map. The parameters include: • *Kb/s*: Configures a specific value in kilobits per second (from 100 to 100000000). • **percent**: Allocates minimum bandwidth to a particular class based on a percentage. The queue can oversubscribe bandwidth in case other queues do not utilize the entire port bandwidth. The total sum cannot exceed 100 percent, and in case it is less than 100 percent, the rest of the bandwidth is equally divided along all bandwidth queues. • **remaining**: Allocates minimum bandwidth to a particular class. The queue can oversubscribe bandwidth in case other queues do not utilize entire port bandwidth. The total sum cannot exceed 100 percent. It is preferred to use this command when the **priority** command is used for certain queues in the policy. You can also assign ratios rather than percentages to each queue; the queues will be assigned certain weights which are inline with these ratios. Ratios can range from 1 to 65536. Total bandwidth ratio allocation for the policy in this case can exceed 100. **Note** You cannot mix bandwidth types on a policy map.
Step 5	**shape average** {*target bit rate* | **percent** *percentage*} **Example:**	Configures the average shape rate. You can configure the average shape rate by target bit rates (bits per second) or by percentage of

	Command or Action	Purpose
	`Device(config-pmap-c)# shape average percent 50` `Device(config-pmap-c)#`	interface bandwidth for the Committed Information Rate (CIR).
Step 6	**end** **Example:** `Device(config-pmap-c)# end` `Device#`	Saves configuration changes.

Sharped Profile Queuing Configuration

The following is the example for sharped queuing:

```
Policy Map test
    Class test1
      bandwidth 20 (%)
      Average Rate Traffic Shaping
      cir 40%
    Class test3
      Average Rate Traffic Shaping
      cir 50%
    Class test2
      Average Rate Traffic Shaping
      cir 50%
    Class test4
      bandwidth 20 (%)
    Class test5
      Average Rate Traffic Shaping
      cir 70%
    Class test6
      Average Rate Traffic Shaping
      cir 60%
```

Monitoring QoS

The following commands can be used to monitor QoS on the device:

Table 138: Monitoring QoS

Command	Description
show class-map [*class_map_name*]	Displays a list of all class maps configured.

Command	Description
show policy-map [*policy_map_name*]	Displays a list of all policy maps configured. Command parameters include: • policy map name • interface • session
show policy-map session [**input** | **output** | **uid** *UUID*]	Displays the session QoS policy. Command parameters include: • input—Input policy • output—Output policy • uid—Policy based on SSS unique identification.
show table-map	Displays all the table maps and their configurations.

Configuration Examples for QoS

The following sections provide configuration examples for QoS.

Examples: TCP Protocol Classification

TCP packets can be classified based on port numbers. The configuration for TCP protocol is as follows:

```
Device#show ip acce tcp
Extended IP access list tcp
    10 permit tcp any any eq 80
Device #
Device #show run class-map tcp

Current configuration : 63 bytes
!
class-map match-all tcp
 match access-group name tcp
!
end
Device #
Device #show run policy-map tcp

Current configuration : 56 bytes
!
policy-map tcp
 class tcp
  police 1000000000
!
end
Device #
```

```
Device #show run  gigabitethernet 1/1

Current configuration : 93 bytes
!
interface gigabitethernet 1/1
 no ip address
 no keepalive
 service-policy output tcp
end

Device #
```

Examples: UDP Protocol Classification

UDP packets can be classified based on port numbers. The configuration example for UDP protocol is as follows:

```
Device#show ip acce udp
Extended IP access list udp
    10 permit udp any any eq ntp
Device #

Device #show run class-map udp
Building configuration...

Current configuration : 63 bytes
!
class-map match-all udp
 match access-group name udp
!
end

Device #
Device #show run policy-map udp
Building configuration...

Current configuration : 56 bytes
!
policy-map udp
 class udp
  police 1000000000
!
end
Device #
Device #show run int gigabitethernet 1/1

Current configuration : 93 bytes
!
interface gigabitethernet 1/1
 no ip address
 no keepalive
 service-policy output udp
end

Device #
```

Examples: RTP Protocol Classification

RTP packets can be classified based on port numbers. The configuration example for RTP protocol is as follows:

```
Device# show ip access-list rtp
Extended IP access list rtp
    10 permit udp any any eq 554
    11 permit tcp any any eq 554
Device #

Device #show run class-map rtp

Current configuration : 63 bytes
!
class-map match-all rtp
 match access-group name rtp
!
end

Device #
Device #show run policy-map rtp

Current configuration : 56 bytes
!
policy-map rtp
 class rtp
  police 1000000000
!
end

Device #
Device #show run gigabitethernet 1/1

Current configuration : 93 bytes
!
interface gigabitethernet 1/1
 no ip address
 no keepalive
 service-policy output rtp
end

Device #
```

Examples: Classification by Access Control Lists

This example shows how to classify packets for QoS by using access control lists (ACLs):

```
Device# configure terminal
Device(config)# access-list 101 permit ip host 12.4.1.1 host 15.2.1.1
Device(config)# class-map acl-101
Device(config-cmap)# description match on access-list 101
Device(config-cmap)# match access-group 101
Device(config-cmap)#
```

After creating a class map by using an ACL, you then create a policy map for the class, and apply the policy map to an interface for QoS.

Examples: Class of Service Layer 2 Classification

This example shows how to classify packets for QoS using a class of service Layer 2 classification:

```
Device# configure terminal
```

```
Device(config)# class-map cos
Device(config-cmap)# match cos ?
  <0-7>  Enter up to 4 class-of-service values separated by white-spaces
Device(config-cmap)# match cos 3 4 5
Device(config-cmap)#
```

After creating a class map by using a CoS Layer 2 classification, you then create a policy map for the class, and apply the policy map to an interface for QoS.

Examples: Class of Service DSCP Classification

This example shows how to classify packets for QoS using a class of service DSCP classification:

```
Device# configure terminal
Device(config)# class-map dscp
Device(config-cmap)# match dscp af21 af22 af23
Device(config-cmap)#
```

After creating a class map by using a DSCP classification, you then create a policy map for the class, and apply the policy map to an interface for QoS.

Examples: VLAN ID Layer 2 Classification

This example shows how to classify for QoS using a VLAN ID Layer 2 classification:

```
Device# configure terminal
Device(config)# class-map vlan-120
Device(config-cmap)# match vlan ?
  <1-4095>  VLAN id
Device(config-cmap)# match vlan 120
Device(config-cmap)#
```

After creating a class map by using a VLAN Layer 2 classification, you then create a policy map for the class, and apply the policy map to an interface for QoS.

Examples: Classification by DSCP or Precedence Values

This example shows how to classify packets by using DSCP or precedence values:

```
Device# configure terminal
Device(config)# class-map prec2
Device(config-cmap)# description matching precedence 2 packets
Device(config-cmap)# match ip precedence 2
Device(config-cmap)# exit
Device(config)# class-map ef
Device(config-cmap)# description EF traffic
Device(config-cmap)# match ip dscp ef
Device(config-cmap)#
```

After creating a class map by using a DSCP or precedence values, you then create a policy map for the class, and apply the policy map to an interface for QoS.

Examples: Hierarchical Classification

The following is an example of a hierarchical classification, where a class named parent is created, which matches another class named child. The class named child matches based on the IP precedence being set to 2.

```
Device# configure terminal
Device(config)# class-map child
Device(config-cmap)# match ip precedence 2
Device(config-cmap)# exit
Device(config)# class-map parent
Device(config-cmap)# match class child
Device(config-cmap)#
```

After creating the parent class map, you then create a policy map for the class, and apply the policy map to an interface for QoS.

Examples: Hierarchical Policy Configuration

The following is an example of a configuration using hierarchical polices:

```
Device# configure terminal
Device(config)# class-map c1
Device(config-cmap)# match dscp 30
Device(config-cmap)# exit

Device(config)# class-map c2
Device(config-cmap)# match precedence 4
Device(config-cmap)# exit

Device(config)# class-map c3
Device(config-cmap)# exit

Device(config)# policy-map child
Device(config-pmap)# class c1
Device(config-pmap-c)# priority level 1
Device(config-pmap-c)# police rate percent 20 conform-action transmit exceed action drop
Device(config-pmap-c-police)# exit
Device(config-pmap-c)# exit

Device(config-pmap)# class c2
Device(config-pmap-c)# bandwidth 20000
Device(config-pmap-c)# exit
Device(config-pmap)# class class-default
Device(config-pmap-c)# bandwidth 20000
Device(config-pmap-c)# exit
Device(config-pmap)# exit

Device(config)# policy-map parent
Device(config-pmap)# class class-default
Device(config-pmap-c)# shape average 1000000
Device(config-pmap-c)# service-policy child
Device(config-pmap-c)# end
```

The following example shows a hierarchical policy using table maps:

```
Device(config)# table-map dscp2dscp
Device(config-tablemap)# default copy
Device(config)# policy-map ssid_child_policy
Device(config-pmap)# class voice
Device(config-pmap-c)# priority level 1
Device(config-pmap-c)# police 15000000
Device(config-pmap)# class video
Device(config-pmap-c)# priority level 2
Device(config-pmap-c)# police 10000000
Device(config)# policy-map ssid_policy
Device(config-pmap)# class class-default
Device(config-pmap-c)# shape average 30000000
Device(config-pmap-c)# queue-buffer ratio 0
Device(config-pmap-c)# set dscp dscp table dscp2dscp
Device(config-pmap-c)# service-policy ssid_child_policy
```

Examples: Classification for Voice and Video

This example describes how to classify packet streams for voice and video using device specific information.

In this example, voice and video are coming in from end-point A into gigabitethernet 1/1 on the device and have precedence values of 5 and 6, respectively. Additionally, voice and video are also coming from end-point B into gigabitethernet 1/1 on the device with DSCP values of EF and AF11, respectively.

Assume that all the packets from the both the interfaces are sent on the uplink interface, and there is a requirement to police voice to 100 Mbps and video to 150 Mbps.

To classify per the above requirements, a class to match voice packets coming in on gigabitethernet 1/1 is created, named voice-interface-1, which matches precedence 5. Similarly another class for voice is created, named voice-interface-2, which will match voice packets in gigabitethernet 1/1. These classes are associated to two separate policies named input-interface-1, which is attached to gigabitethernet 1/1, and input-interface-2, which is attached to gigabitethernet 1/1. The action for this class is to mark the qos-group to 10. To match packets with QoS-group 10 on the output interface, a class named voice is created which matches on QoS-group 10. This is then associated to another policy named output-interface, which is associated to the uplink interface. Video is handled in the same way, but matches on QoS-group 20.

The following example shows how classify using the above device specific information:

```
Device(config)#
Device(config)# class-map voice-interface-1
Device(config-cmap)# match ip precedence 5
Device(config-cmap)# exit

Device(config)# class-map video-interface-1
Device(config-cmap)# match ip precedence 6
Device(config-cmap)# exit

Device(config)# class-map voice-interface-2
Device(config-cmap)# match ip dscp ef
Device(config-cmap)# exit

Device(config)# class-map video-interface-2
Device(config-cmap)# match ip dscp af11
Device(config-cmap)# exit

Device(config)# policy-map input-interface-1
Device(config-pmap)# class voice-interface-1
Device(config-pmap-c)# set qos-group 10
```

```
Device(config-pmap-c)# exit

Device(config-pmap)# class video-interface-1
Device(config-pmap-c)# set qos-group 20

Device(config-pmap-c)# policy-map input-interface-2
Device(config-pmap)# class voice-interface-2
Device(config-pmap-c)# set qos-group 10
Device(config-pmap-c)# class video-interface-2
Device(config-pmap-c)# set qos-group 20
Device(config-pmap-c)# exit
Device(config-pmap)# exit

Device(config)# class-map voice
Device(config-cmap)# match qos-group 10
Device(config-cmap)# exit

Device(config)# class-map video
Device(config-cmap)# match qos-group 20
Device(config)# policy-map output-interface
Device(config-pmap)# class voice
Device(config-pmap-c)# police 256000 conform-action transmit exceed-action drop
Device(config-pmap-c-police)# exit
Device(config-pmap-c)# exit

Device(config-pmap)# class video
Device(config-pmap-c)# police 1024000 conform-action transmit exceed-action drop
Device(config-pmap-c-police)# exit
Device(config-pmap-c)# exit
```

Examples: Average Rate Shaping Configuration

The following example shows how to configure average rate shaping:

```
Device# configure terminal
Device(config)# class-map prec1
Device(config-cmap)# description matching precedence 1 packets
Device(config-cmap)# match ip precedence 1
Device(config-cmap)# end

Device# configure terminal
Device(config)# class-map prec2
Device(config-cmap)# description matching precedence 2 packets
Device(config-cmap)# match ip precedence 2
Device(config-cmap)# exit

Device(config)# policy-map shaper
Device(config-pmap)# class prec1
Device(config-pmap-c)# shape average 512000
Device(config-pmap-c)# exit

Device(config-pmap)# policy-map shaper
Device(config-pmap)# class prec2
Device(config-pmap-c)# shape average 512000
Device(config-pmap-c)# exit

Device(config-pmap)# class class-default
Device(config-pmap-c)# shape average 1024000
```

After configuring the class maps, policy map, and shape averages for your configuration, proceed to then apply the policy map to the interface for QoS.

Examples: Queue-limit Configuration

The following example shows how to configure a queue-limit policy based upon DSCP values and percentages:

```
Device# configure terminal
Device#(config)# policy-map port-queue
Device#(config-pmap)# class dscp-1-2-3
Device#(config-pmap-c)# bandwidth percent 20
Device#(config-pmap-c)# queue-limit dscp 1 percent 80
Device#(config-pmap-c)# queue-limit dscp 2 percent 90
Device#(config-pmap-c)# queue-limit dscp 3 percent 100
Device#(config-pmap-c)# exit

Device#(config-pmap)# class dscp-4-5-6
Device#(config-pmap-c)# bandwidth percent 20
Device#(config-pmap-c)# queue-limit dscp 4 percent 20
Device#(config-pmap-c)# queue-limit dscp 5 percent 30
Device#(config-pmap-c)# queue-limit dscp 6 percent 20
Device#(config-pmap-c)# exit

Device#(config-pmap)# class dscp-7-8-9
Device#(config-pmap-c)# bandwidth percent 20
Device#(config-pmap-c)# queue-limit dscp 7 percent 20
Device#(config-pmap-c)# queue-limit dscp 8 percent 30
Device#(config-pmap-c)# queue-limit dscp 9 percent 20
Device#(config-pmap-c)# exit

Device#(config-pmap)# class dscp-10-11-12
Device#(config-pmap-c)# bandwidth percent 20
Device#(config-pmap-c)# queue-limit dscp 10 percent 20
Device#(config-pmap-c)# queue-limit dscp 11 percent 30
Device#(config-pmap-c)# queue-limit dscp 12 percent 20
Device#(config-pmap-c)# exit

Device#(config-pmap)# class dscp-13-14-15
Device#(config-pmap-c)# bandwidth percent 10
Device#(config-pmap-c)# queue-limit dscp 13 percent 20
Device#(config-pmap-c)# queue-limit dscp 14 percent 30
Device#(config-pmap-c)# queue-limit dscp 15 percent 20
Device#(config-pmap-c)# end
Device#
```

After finishing with the above policy map queue-limit configuration, you can then proceed to apply the policy map to an interface for QoS.

Examples: Queue Buffers Configuration

The following example shows how configure a queue buffer policy and then apply it to an interface for QoS:

```
Device# configure terminal
Device(config)# policy-map policy1001
Device(config-pmap)# class class1001
Device(config-pmap-c)# bandwidth remaining ratio 10
Device(config-pmap-c)# queue-buffer ratio ?
  <0-100>  Queue-buffers ratio limit
```

```
Device(config-pmap-c)# queue-buffer ratio 20
Device(config-pmap-c)# end

Device# configure terminal
Device(config)# interface gigabitethernet 1/1
Device(config-if)# service-policy output policy1001
Device(config-if)# end
```

Examples: Policing Action Configuration

The following example displays the various policing actions that can be associated to the policer. These actions are accomplished using the conforming, exceeding, or violating packet configurations. You have the flexibility to drop, mark and transmit, or transmit packets that have exceeded or violated a traffic profile.

For example, a common deployment scenario is one where the enterprise customer polices traffic exiting the network towards the service provider and marks the conforming, exceeding and violating packets with different DSCP values. The service provider could then choose to drop the packets marked with the exceeded and violated DSCP values under cases of congestion, but may choose to transmit them when bandwidth is available.

Note The Layer 2 fields can be marked to include the CoS fields, and the Layer 3 fields can be marked to include the precedence and the DSCP fields.

One useful feature is the ability to associate multiple actions with an event. For example, you could set the precedence bit and the CoS for all conforming packets. A submode for an action configuration could then be provided by the policing feature.

This is an example of a policing action configuration:

```
Device# configure terminal
Device(config)# policy-map police
Device(config-pmap)# class class-default
Device(config-pmap-c)# police cir 1000000 pir 2000000
Device(config-pmap-c-police)# conform-action transmit
Device(config-pmap-c-police)# exceed-action set-dscp-transmit dscp table exceed-markdown-table
Device(config-pmap-c-police)# violate-action set-dscp-transmit dscp table
violate-markdown-table
Device(config-pmap-c-police)# end
```

In this example, the exceed-markdown-table and violate-mark-down-table are table maps.

Note Policer-based markdown actions are only supported using table maps. Only one markdown table map is allowed for each marking field in the device.

Examples: Policer VLAN Configuration

The following example displays a VLAN policer configuration. At the end of this configuration, the VLAN policy map is applied to an interface for QoS.

```
Device# configure terminal
Device(config)# class-map vlan100
Device(config-cmap)# match vlan 100
Device(config-cmap)# exit
Device(config)# policy-map vlan100
Device(config-pmap)# policy-map class vlan100
Device(config-pmap-c)# police 100000 bc conform-action transmit exceed-action drop
Device(config-pmap-c-police)# end
Device# configure terminal
Device(config)# interface gigabitethernet 1/1
Device(config-if)#  service-policy input vlan100
```

Examples: Policing Units

The policing unit is the basis on which the token bucket works. CIR and PIR are specified in bits per second. The burst parameters are specified in bytes. This is the default mode; it is the unit that is assumed when no units are specified. The CIR and PIR can also be configured in percent, in which case the burst parameters have to be configured in milliseconds.

The following is an example of a policer configuration in bits per second. In this configuration, a dual-rate three-color policer is configured where the units of measurement is bits. The burst and peak burst are all specified in bits.

```
Device(config)# policy-map bps-policer
Device(config-pmap)# class class-default
Device(config-pmap-c)# police rate 100000 peak-rate 1000000
conform-action transmit exceed-action set-dscp-transmit dscp table
DSCP_EXCE violate-action drop
```

Examples: Single-Rate Two-Color Policing Configuration

The following example shows how to configure a single-rate two-color policer:

```
Device(config)# class-map match-any prec1
Device(config-cmap)# match ip precedence 1
Device(config-cmap)# exit
Device(config)# policy-map policer
Device(config-pmap)# class prec1
Device(config-pmap-c)# police cir 256000 conform-action transmit exceed-action drop
Device(config-pmap-c-police)# exit
Device(config-pmap-c)#
```

Examples: Dual-Rate Three-Color Policing Configuration

The following example shows how to configure a dual-rate three-color policer:

```
Device# configure terminal
Device(config)# policy-Map dual-rate-3color-policer
Device(config-pmap)# class class-default
Device(config-pmap-c)# police cir 64000 bc 2000 pir 128000 be 2000
Device(config-pmap-c-police)# conform-action transmit
```

```
Device(config-pmap-c-police)# exceed-action set-dscp-transmit dscp table exceed-markdown-table
Device(config-pmap-c-police)# violate-action set-dscp-transmit dscp table
violate-markdown-table
Device(config-pmap-c-police)# exit
Device(config-pmap-c)#
```

In this example, the exceed-markdown-table and violate-mark-down-table are table maps.

Note Policer based markdown actions are only supported using table maps. Only one markdown table map is allowed for each marking field in the device.

Examples: Table Map Marking Configuration

The following steps and examples show how to use table map marking for your QoS configuration:

1. Define the table map.

   Define the table-map using the **table-map** command and indicate the mapping of the values. This table does not know of the policies or classes within which it will be used. The default command in the table map indicates the value to be copied into the 'to' field when there is no matching 'from' field. In the example, a table map named table-map1 is created. The mapping defined is to convert the value from 0 to 1 and from 2 to 3, while setting the default value to 4.

   ```
   Device(config)# table-map table-map1
   Device(config-tablemap)# map from 0 to 1
   Device(config-tablemap)# map from 2 to 3
   Device(config-tablemap)# default 4
   Device(config-tablemap)# exit
   ```

2. Define the policy map where the table map will be used.

   In the example, the incoming CoS is mapped to the DSCP based on the mapping specified in the table table-map1. For this example, if the incoming packet has a DSCP of 0, the CoS in the packet is set 1. If no table map name is specified the command assumes a default behavior where the value is copied as is from the 'from' field (DSCP in this case) to the 'to' field (CoS in this case). Note however, that while the CoS is a 3-bit field, the DSCP is a 6-bit field, which implies that the CoS is copied to the first three bits in the DSCP.

   ```
   Device(config)# policy map policy1
   Device(config-pmap)# class class-default
   Device(config-pmap-c)# set cos dscp table table-map1
   Device(config-pmap-c)# exit
   ```

3. Associate the policy to an interface.

   ```
   Device(config)# interface gigabitethernet 1/1
   Device(config-if)# service-policy output policy1
   Device(config-if)# exit
   ```

Example: Table Map Configuration to Retain CoS Markings

The following example shows how to use table maps to retain CoS markings on an interface for your QoS configuration.

The cos-trust-policy policy (configured in the example) is enabled in the ingress direction to retain the CoS marking coming into the interface. If the policy is not enabled, only the DSCP is trusted by default. If a pure Layer 2 packet arrives at the interface, then the CoS value will be rewritten to 0 when there is no such policy in the ingress port for CoS.

```
Device# configure terminal
Device(config)# table-map cos2cos
Device(config-tablemap)# default copy
Device(config-tablemap)# exit


Device(config)# policy map cos-trust-policy
Device(config-pmap)# class class-default
Device(config-pmap-c)# set cos cos table cos2cos
Device(config-pmap-c)# exit

Device(config)# interface gigabitethernet 1/1
Device(config-if)# service-policy input cos-trust-policy
Device(config-if)# exit
```

Where to Go Next

Review the auto-QoS documentation to see if you can use these automated capabilities for your QoS configuration.

CHAPTER 133

Configuring Weighted Random Early Detection

Avoiding Network Congestion

Heterogeneous networks include different protocols used by applications, giving rise to the need to prioritize traffic in order to satisfy time-critical applications while still addressing the needs of less time-dependent applications, such as file transfer. If your network is designed to support different traffic types that share a single data path between devices in a network, implementing congestion avoidance mechanisms ensures fair treatment across the various traffic types and avoids congestion at common network bottlenecks. Congestion avoidance mechanism is achieved through packet dropping.

Random Early Detection (RED) is a commonly used congestion avoidance mechanism in a network.

Tail Drop

Tail drop treats all traffic equally and does not differentiate within a class of service. When the output queue is full and tail drop is in effect, packets are dropped until the congestion is eliminated and the queue is no longer full.

Weighted Random Early Detection

The RED mechanism takes advantage of the congestion control mechanism of TCP. Packets are randomly dropped prior to periods of high congestion. Assuming the packet source uses TCP, it decreases its transmission rate until all the packets reach their destination, indicating that the congestion is cleared. You can use RED

as a way to cause TCP to slow down transmission of packets. TCP not only pauses, but also restarts quickly and adapts its transmission rate to the rate that the network can support.

WRED is the Cisco implementation of RED. It combines the capabilities of RED algorithm with IP Precedence or Differentiated Services Code Point (DSCP) or Class of Service (COS) values.

How WRED Works

WRED reduces the chances of tail drop by selectively dropping packets when the output interface begins to show signs of congestion. WRED drops some packets early rather than waiting until the queue is full. Thus it avoids dropping large number of packets at once and minimizes the chances of TCP global synchronization.

Approximate Fair Drop (AFD) is an Active Queue Management (AQM) algorithm that determines the packet drop probability. The probability of dropping packets depends upon the arrival rate calculation of a flow at ingress and the current queue length.

AFD based WRED is implemented on wired network ports.

AFD based WRED emulates the preferential dropping behavior of WRED. This preferential dropping behavior is achieved by changing the weights of AFD sub-classes based on their corresponding WRED drop thresholds. Within a physical queue, traffic with larger weight incurs less drop probability than that of smaller weight.

- Each WRED enabled queue has high and low thresholds.
- A sub-class of higher priority has a larger AFD weight.
- The sub-classes are sorted in ascending order, based on lowest of WRED minThreshold.

WRED Weight Calculation

AFD weight is calculated using low and high threshold values; AFD is an adjsuted index of the average of WRED high and WRED low threshold values.

When a packet arrives at an interface, the following events occur:

1. The drop probability is calculated. The drop probability increases as the AFD weight decreases. That means, if the average of low and high threshold values is less, the drop probability is more.
2. WRED consideres the priority of packet flows and the threshold values before deciding to drop the packet. The CoS, DSCP or IP Precedence values are mapped to the specified thresholds. Once these thresholds are exceeded, packets with the configured values that are mapped to these thresholds are eligible to be dropped. Other packets with CoS, DSCP or IP Precedence values assigned to the higher thresholds are en-queued. This process keeps the higher priority flows intact and minimizes the latency in packet transmission.
3. If packets are not dropped using WRED, they are tail-dropped.

Limitations for WRED Configuration

- Weighted Tail Drop (WTD) is enabled by default on all the queues.
- WRED can be enabled / disabled per queue. When WRED is disabled, WTD is adapted on the target queue. Policy-map with WRED profile is configured only on physical ports as output policy.

- WRED is supported only in network port queues and is not supported on internal CPU queues.
- Each WRED physical queue can support three different threshold pairs. Each pair is for one QoS tag value.
- Ensure that you configure bandwidth or shape in the policy-map along with WRED.
- Specify all the WRED thresholds only in percentage mode.
- Map the WRED threshold pairs by mapping class-map filter with corresponding match filters.

  We recommend the class-map with match “any” filter.
- WRED for priority traffic is not supported.
- WRED and queue limit are not supported for the same policy.
- Wired ports support a maximum of eight physical queues, of which you can configure WRED only on four physical queues, each with three threshold pairs. The remaining queues are configured with WTD. Policies with more than four WRED queues are rejected.

Usage Guidelines for WRED

To configure AFD based WRED feature, specify the policy map and add the class. Use the **random-detect** command to specify the method (using the dscp-based / cos-based / precedence-based arguments) that you want WRED to use to calculate the drop probability.

Note You can modify the policy on the fly. The AFD weights are automatically recalculated.

WRED can be configured for any kind of traffic like IPv4/IPv6, Multicast, and so on. WRED is supported on all 8 queueing classes.

Consider the following points when you are configuring WRED with **random-detect** command:

- With dscp-based argument, WRED uses the DSCP value to calculate drop probability.
- With cos-based argument, WRED uses the COS value to calculate drop probability.
- By default, WRED uses the IP Precedence value to calculate drop probability. **precedence-based** argument is the default and it is not displayed in the CLI.

Note **show run policy-map** *policy-map* command does not display “precedence” though precedence is configured with **random-detect** command.

- The dscp-based and precedence-based arguments are mutually exclusive.
- Each of the eight physical queues can be configured with different WRED profiles.

Configuring WRED

Configuring WRED based on DSCP Values

Use the following steps to configure WRED profile based on DSCP values in packet mode:

Procedure

	Command or Action	Purpose
Step 1	**class-map** *match-criteria class-name* **Example:** `device(config)# class-map match-any CS`	Configures match criteria for the class map. Recommended match-criteria is match-any
Step 2	**match** *class-map-name* **Example:** `device(config-cmap)#match dscp CS1`	Match a class-map.
Step 3	**policy-map** *name* **Example:** `device(config)#policy-map PWRED`	Specifies the name of the WRED profile policy to be created.
Step 4	**class** *class-name* **Example:** `device(config-pmap)#class CS`	Specifies the name of the Class to be associated with the policy.
Step 5	Use either **bandwidth** {*kbps* | **remaining ratio** | **percent** *percentage*} or **shape** { **average** | **peak** } *cir* **Example:** `device(config-pmap-c)#bandwidth percent 10`	Specify either the bandwidth allocated for a class belonging to a policy map or the traffic shaping.
Step 6	**random-detect** *dscp-based* **Example:** `device(config-pmap-c)#random-detect dscp-based`	Configures WRED to use the DSCP value when it calculates the drop probability for the packet.
Step 7	**random-detect dscp** *dscp-value* **percent** *minThreshold maxThreshold* **Example:** `device(config-pmap-c)#random-detect dscp cs1 percent 10 20`	Specifies the minimum and maximum thresholds, in percentage.

	Command or Action	Purpose
Step 8	**interface** *interface-name* **Example:** `device(config)#interface gigabitethernet 1/1`	Enters the interface configuration mode.
Step 9	**service-policy output** *policy-map* **Example:** `device(config-if)#service-policy output pwred`	Attaches the policy map to an output interface.

Configuring WRED based on Class of Service Values

Use the following steps to configure WRED profile based on Class of Service (COS) values in packet mode:

Procedure

	Command or Action	Purpose
Step 1	**class-map** *match-criteria class-name* **Example:** `device(config)# class-map match-any CS`	Configures match criteria for the class map. Recommended match-criteria is match-any
Step 2	**match** *class-map-name* **Example:** `device(config-cmap)#match cos 3`	Match a class-map.
Step 3	**policy-map** *name* **Example:** `device(config)#policy-map PWRED`	Specifies the name of the WRED profile policy to be created.
Step 4	**class** *class-name* **Example:** `device(config-pmap)#class CS`	Specifies the name of the Class to be associated with the policy.
Step 5	**bandwidth** {*kbps*| **remaining** *percentage* | **percent** *percentage*} **Example:** `device(config-pmap-c)#bandwidth percent 10`	Specifies the bandwidth allocated for a class belonging to a policy map.
Step 6	**random-detect** *cos-based* **Example:** `device(config-pmap-c)#random-detect cos-based`	Configures WRED to use the CoS value when it calculates the drop probability for the packet.

	Command or Action	Purpose
Step 7	**random-detect cos** *cos-value* **percent** *minThreshold maxThreshold* **Example:** `device(config-pmap-c)#random-detect cos 3 percent 10 20`	Specifies the minimum and maximum thresholds, in percentage.
Step 8	**interface** *interface-name* **Example:** `device(config)# interface gigabitethernet 1/1`	Enters the interface configuration mode.
Step 9	**service-policy output** *policy-map* **Example:** `device(config-if)#service-policy output pwred`	Attaches the policy map to an output interface.

Configuring WRED based on IP Precedence Values

Use the following steps to configure WRED profile based on IP precedence values in packet mode:

Procedure

	Command or Action	Purpose
Step 1	**class-map** *match-criteria class-name* **Example:** `device(config)# class-map match-any CS`	Configures match criteria for the class map. Recommended match-criteria is match-any
Step 2	**match** *class-map-name* **Example:** `device(config-cmap)#match precedence 3`	Match a class-map.
Step 3	**policy-map** *name* **Example:** `device(config)#policy-map pwred`	Specifies the name of the WRED profile policy to be created.
Step 4	**class** *class-name* **Example:** `device(config-pmap)#class CS`	Specifies the name of the Class to be associated with the policy.
Step 5	**bandwidth** {*kbps*| **remaining** *percentage* | **percent** *percentage*} **Example:** `device(config-pmap-c)#bandwidth percent 10`	Specifies the bandwidth allocated for a class belonging to a policy map.

	Command or Action	Purpose
Step 6	**random-detect** *precedence-based* **Example:** `device(config-pmap-c)#random-detect precedence-based`	Configures WRED to use the IP precedence value when it calculates the drop probability for the packet.
Step 7	**random-detect precedence** *precedence-value* **percent** *minThreshold maxThreshold* **Example:** `device(config-pmap-c)#random-detect precedence 3 percent 10 20`	Specifies the minimum and maximum thresholds, in percentage.
Step 8	**interface** *interface-name* **Example:** `device(config)#interface gigabitethernet 1/1`	Enters the interface configuration mode.
Step 9	**service-policy output** *policy-map* **Example:** `device(config-if)#service-policy output pwred`	Attaches the policy map to an output interface.

WRED Configuration Example

WRED Support with Hierarchical QoS

Hierarchical QoS allows you to specify QoS behavior at multiple policy levels, which provides a high degree of granularity in traffic management.

For HQoS, WRED is allowed only on the child policy and not on the parent policy. You can have the shaping configured on the parent policy and WRED on the child.

The following example configures the parent policy **pwred-parent** with traffic shaped on the basis of 10 percent of the bandwidth, that applies to its child, **pwred-child** configured for DSCP-based WRED.

```
policy-map PWRED-CHILD
 class CWRED
    bandwidth percent 10
    random-detect dscp-based
    random-detect dscp 1 percent 10 20
    random-detect dscp 10 percent 20 30

policy-map PWRED-PARENT
  class class-default
  shape average percent 10
  service-policy PWRED-CHILD
```

Displaying WRED Configuration

The following example shows how to display the WRED and threshold labels:

```
Device# show policy-map PWRED

  Policy Map PWRED
    Class CS
      bandwidth 10 (%)
       percent-based wred

      dscp    min-threshold    max-threshold
      ------------------------------------
      cs1 (8)      10               20
      cs2 (16)     20               30
      cs3 (24)     34               44
      default (0)   -
```

The following example shows how to display WRED AFD Weights, WRED Enq (in Packets and Bytes), WRED Drops (in Packets and Bytes), Configured DSCP labels against the Threshold pairs:

Note Use this command only after you initiate the traffic. **show policy-map interface** is updated with WRED configuration only after a traffic is sent.

```
Device# show policy-map interface gigabitethernet 1/1

gigabitethernet 1/1

  Service-policy output: PWRED

    Class-map: CS (match-any)
      0 packets
      Match:  dscp cs1 (8)
      Match:  dscp cs2 (16)
      Match:  dscp cs3 (24)
      Queueing

      (total drops) 27374016
      (bytes output) 33459200081
      bandwidth 10% (1000000 kbps)

AFD WRED STATS BEGIN
Virtual Class   min/max        Transmit                    Random drop                AFD
 Weight


   0          10 / 20        (Byte)33459183360           27374016                    12
                             (Pkts)522799759                427716
              dscp : 8


   1          20 / 30          Byte)0                      0                         20
                                (Pkts)0                    0
              dscp : 16


   2          34 / 44        (Byte)16721                  0                         31
                               (Pkts)59                    0
```

```
          dscp : 24

     Total Drops(Bytes)   : 27374016

     Total Drops(Packets) : 427716
AFD WRED STATS END


    Class-map: class-default (match-any)
      0 packets
      Match: any


      (total drops) 0
      (bytes output) 192
```

Best Practices for WRED Configuration

- **Support for three WRED configuration pairs**

  Each WRED Physical Queue (AFD Queue) can support three WRED configuation pairs, with unique WRED threshold pair configuration.

```
Policy-map P1
    Class CS
        Random-detect dscp-based
        Random-detect dscp CS1 percent 10 20      // WRED pair 1
        Random-detect dscp CS2  percent 20 30      // WRED pair 2
        Random-detect dscp CS3 percent 30 40      // WRED pair 3
Class-map match-any CS
        match cs1
        match cs2
        match cs3
```

- **Appending WRED configuration pairs**

  You can add overlapping threshold pairs into the WRED configuration pairs.

```
Policy-map P1
    Class CS
       Random-detect dscp-based
       Random-detect dscp CS1 percent 10 20       // WRED pair 1
       Random-detect dscp CS2 percent 20 30       // WRED pair 2
       Random-detect dscp CS3 percent 30 40       // WRED pair 3
       Random-detect dscp CS4 percent  30 40     ==> belongs to WRED pair 3
       Random-detect dscp CS5 percent 20 30     ==> belongs to WRED pair 2
Class-map match-any CS
       match cs1
       match cs2
       match cs3
       match cs4  >>
       match cs5  >>
```

- **Default WRED pairs**

  If less than three WRED pairs are configured, any class-map filter participating WRED gets assigned to the third default WRED pair with maximum threshold (100, 100).

```
Policy-map P1
    Class CS
        Random-detect dscp-based
        Random-detect dscp CS1 percent 10 20   // WRED pair 1
```

```
        Random-detect dscp CS2 percent 20 30   // WRED pair 2
Class-map match-any CS
        match CS1
        match CS2
        match CS3
        match CS4
```

In this case, classes CS3 and CS4 are mapped to WRED pair 3 with threshold (100, 100).

- **Rejection of Mismatched Configuration**

  If you configure random-detect without matching filters in a class-map, the policy installation is rejected.

```
Class-map match-any CS
       match CS1
       match CS2
       match CS5
Policy-map P1
    Class CS
        Shape average percent 10
        Random-detect dscp-based
        Random-detect dscp CS1 percent 10 20   // WRED pair 1
        Random-detect dscp CS2 percent 20 30   // WRED pair 2
       Random-detect dscp CS3 percent 30 40   // WRED pair 3  ===> Mismatched sub-class.
```

  When this policy is applied to the interface on the egress side, the policy fails during installation as the class-map values are incorrect:

```
device(config)# int gigabitethernet 1/1
device(config-if)# service-policy output P1
device(config-if)#
*Feb 20 17:33:16.964: %IOSXE-5-PLATFORM: Switch 1 R0/0: fed: WRED POLICY INSTALL
FAILURE.Invalid WRED filter mark: 24 in class-map: CS
*Feb 20 17:33:16.965: %FED_QOS_ERRMSG-3-LABEL_2_QUEUE_MAPPING_HW_ERROR: Switch 1 R0/0:
 fed: Failed to detach queue-map for gigabitethernet 1/1: code 2
```

PART VI

System Management

CHAPTER 134

Administering the Device

Information About Administering the Device

The following sections provide information about administering the device:

System Time and Date Management

You can manage the system time and date on your device using automatic configuration methods (RTC and NTP), or manual configuration methods.

Note For complete syntax and usage information for the commands used in this section, see the *Cisco IOS Configuration Fundamentals Command Reference* on *Cisco.com*.

System Clock

The basis of the time service is the system clock. This clock runs from the moment the system starts up and keeps track of the date and time.

The system clock can then be set from these sources:

- RTC
- NTP
- Manual configuration

The system clock can provide time to these services:

- User **show** commands
- Logging and debugging messages

The system clock keeps track of time internally based on Coordinated Universal Time (UTC), also known as Greenwich Mean Time (GMT). You can configure information about the local time zone and summer time (daylight saving time) so that the time appears correctly for the local time zone.

The system clock keeps track of whether the time is *authoritative* or not (that is, whether it has been set by a time source considered to be authoritative). If it is not authoritative, the time is available only for display purposes and is not redistributed.

Real Time Clock

A real-time clock (RTC) keeps track of the current time on the switch. The switch is shipped to you with RTC set to GMT time until you reconfigure clocking parameters.

The benefits of an RTC are:

- RTC is battery-powered.
- System time is retained during power outage and at system reboot.

The RTC and NTP clocks are integrated on the switch. When NTP is enabled, the RTC time is periodically synchronized to the NTP clock to maintain accuracy.

Network Time Protocol

The NTP is designed to time-synchronize a network of devices. NTP runs over User Datagram Protocol (UDP), which runs over IP. NTP is documented in RFC 1305. The current protocol is version 4 (NTPv4), which is a proposed standard as documented in RFC 5905. It is backward compatible with version 3, specified in RFC 1305.

An NTP network usually gets its time from an authoritative time source, such as a radio clock or an atomic clock attached to a time server. NTP then distributes this time across the network. NTP is extremely efficient; no more than one packet per minute is necessary to synchronize two devices to within a millisecond of one another.

NTP uses the concept of a *stratum* to describe how many NTP hops away a device is from an authoritative time source. A stratum 1 time server has a radio or atomic clock directly attached, a stratum 2 time server receives its time through NTP from a stratum 1 time server, and so on. A device running NTP automatically chooses as its time source the device with the lowest stratum number with which it communicates through NTP. This strategy effectively builds a self-organizing tree of NTP speakers.

NTP avoids synchronizing to a device whose time might not be accurate by never synchronizing to a device that is not synchronized. NTP also compares the time reported by several devices and does not synchronize to a device whose time is significantly different than the others, even if its stratum is lower.

The communications between devices running NTP (known as associations) are usually statically configured; each device is given the IP address of all devices with which it should form associations. Accurate timekeeping is possible by exchanging NTP messages between each pair of devices with an association. However, in a LAN environment, NTP can be configured to use IP broadcast messages instead. This alternative reduces configuration complexity because each device can simply be configured to send or receive broadcast messages. However, in that case, information flow is one-way only.

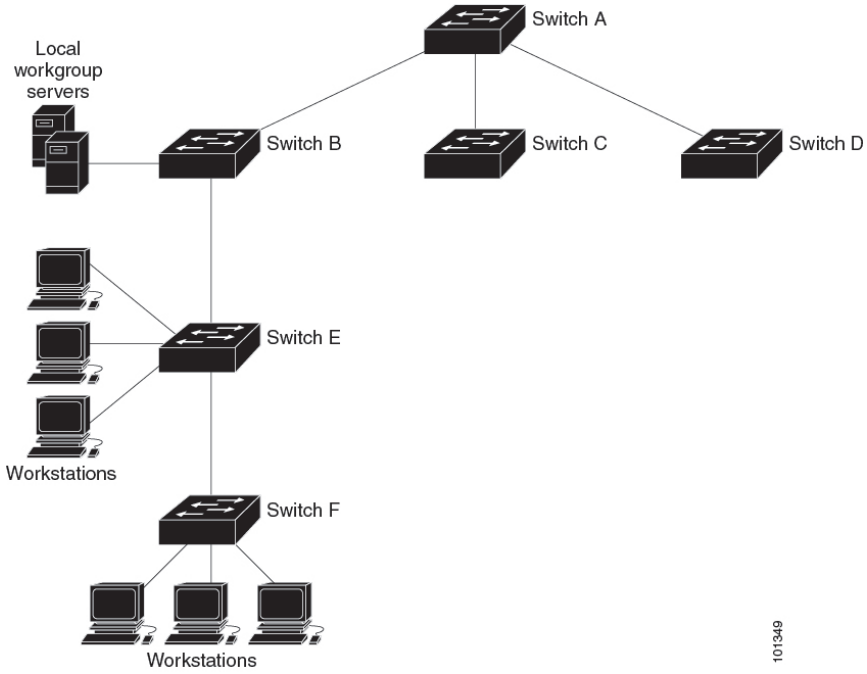
The time kept on a device is a critical resource; you should use the security features of NTP to avoid the accidental or malicious setting of an incorrect time. Two mechanisms are available: an access list-based restriction scheme and an encrypted authentication mechanism.

Cisco's implementation of NTP does not support stratum 1 service; it is not possible to connect to a radio or atomic clock. We recommend that the time service for your network be derived from the public NTP servers available on the IP Internet.

The Figure shows a typical network example using NTP. Device A is the primary NTP, with the **Device** B, C, and D configured in NTP server mode, in server association with Device A. Device E is configured as an NTP peer to the upstream and downstream device, Device B and Device F, respectively.

Figure 141: NTP Network Configuration

An example of a typical network using NTP



If the network is isolated from the Internet, Cisco's implementation of NTP allows a device to act as if it is synchronized through NTP, when in fact it has learned the time by using other means. Other devices then synchronize to that device through NTP.

When multiple sources of time are available, NTP is always considered to be more authoritative. NTP time overrides the time set by any other method.

Several manufacturers include NTP software for their host systems, and a publicly available version for systems running UNIX and its various derivatives is also available. This software allows host systems to be time-synchronized as well.

NTP Stratum

NTP uses the concept of a *stratum* to describe how many NTP hops away a device is from an authoritative time source. A stratum 1 time server has a radio or atomic clock directly attached, a stratum 2 time server receives its time through NTP from a stratum 1 time server, and so on. A device running NTP automatically chooses as its time source the device with the lowest stratum number with which it communicates through NTP. This strategy effectively builds a self-organizing tree of NTP speakers.

NTP avoids synchronizing to a device whose time might not be accurate by never synchronizing to a device that is not synchronized. NTP also compares the time reported by several devices and does not synchronize to a device whose time is significantly different than the others, even if its stratum is lower.

NTP Associations

The communications between devices running NTP (known as *associations*) are usually statically configured; each device is given the IP address of all devices with which it should form associations. Accurate timekeeping is possible by exchanging NTP messages between each pair of devices with an association. However, in a LAN environment, NTP can be configured to use IP broadcast messages instead. This alternative reduces configuration complexity because each device can simply be configured to send or receive broadcast messages. However, in that case, information flow is one-way only.

Poll-Based NTP Associations

Networking devices running NTP can be configured to operate in variety of association modes when synchronizing time with reference time sources. A networking device can obtain time information on a network in two ways—by polling host servers and by listening to NTP broadcasts. This section focuses on the poll-based association modes. Broadcast-based NTP associations are discussed in the *Broadcast-Based NTP Associations* section.

The following are the two most commonly used poll-based association modes:

- Client mode
- Symmetric active mode

The client and the symmetric active modes should be used when NTP is required to provide a high level of time accuracy and reliability.

When a networking device is operating in the client mode, it polls its assigned time-serving hosts for the current time. The networking device will then pick a host from among all the polled time servers to synchronize with. Because the relationship that is established in this case is a client-host relationship, the host will not capture or use any time information sent by the local client device. This mode is most suited for file-server and workstation clients that are not required to provide any form of time synchronization to other local clients. Use the **ntp server** command to individually specify the time server that you want your networking device to consider synchronizing with and to set your networking device to operate in the client mode.

When a networking device is operating in the symmetric active mode, it polls its assigned time-serving hosts for the current time and it responds to polls by its hosts. Because this is a peer-to-peer relationship, the host will also retain time-related information of the local networking device that it is communicating with. This mode should be used when a number of mutually redundant servers are interconnected via diverse network paths. Most stratum 1 and stratum 2 servers on the Internet adopt this form of network setup. Use the **ntp peer** command to individually specify the time serving hosts that you want your networking device to consider synchronizing with and to set your networking device to operate in the symmetric active mode.

The specific mode that you should set for each of your networking devices depends primarily on the role that you want them to assume as a timekeeping device (server or client) and the device's proximity to a stratum 1 timekeeping server.

A networking device engages in polling when it is operating as a client or a host in the client mode or when it is acting as a peer in the symmetric active mode. Although polling does not usually place a burden on memory and CPU resources such as bandwidth, an exceedingly large number of ongoing and simultaneous polls on a system can seriously impact the performance of a system or slow the performance of a given network. To avoid having an excessive number of ongoing polls on a network, you should limit the number of direct, peer-to-peer or client-to-server associations. Instead, you should consider using NTP broadcasts to propagate time information within a localized network.

Broadcast-Based NTP Associations

Broadcast-based NTP associations should be used when time accuracy and reliability requirements are modest and if your network is localized and has more than 20 clients. Broadcast-based NTP associations are also recommended for use on networks that have limited bandwidth, system memory, or CPU resources.

A networking device operating in the broadcast client mode does not engage in any polling. Instead, it listens for NTP broadcast packets that are transmitted by broadcast time servers. Consequently, time accuracy can be marginally reduced because time information flows only one way.

Use the **ntp broadcast client** command to set your networking device to listen for NTP broadcast packets propagated through a network. For broadcast client mode to work, the broadcast server and its clients must be located on the same subnet. You must enable the time server that transmits NTP broadcast packets on the interface of the given device by using the **ntp broadcast** command.

Authoritative NTP Server

An authoritative NTP server is a time server that can distribute time in the network. Other devices can configure it as a time server. You can configure a switch to act as an authoritative NTP server, enabling it to distribute time even when it is not synchronized to an outside time source. Use the **ntp master** command, in global configuration mode, to configure the device to be an authoritative NTP server.

Caution Use the **ntp master** command with caution. Usage of this command can override valid time sources, especially if a low stratum number is configured. Configuring multiple devices in the same network with the **ntp master** command can cause instability in timekeeping if the devices do not agree on the time.

NTP Security

The time kept on a device is a critical resource; you should use the security features of NTP to avoid the accidental or malicious setting of an incorrect time. Two mechanisms are available: an access list-based restriction scheme and an encrypted authentication mechanism.

Note We do not recommend configuring Message Direct 5 (MD5) authentication. You can use other supported authentication methods for stronger encryption.

NTP Access Group

The access list-based restriction scheme allows you to grant or deny certain access privileges to an entire network, a subnet within a network, or a host within a subnet. To define an NTP access group, use the ntp access-group command in global configuration mode.

The access group options are scanned in the following order, from least restrictive to the most restrictive:

1. ipv4 —Configures IPv4 access lists.
2. ipv6 —Configures IPv6 access lists.
3. peer —Allows time requests and NTP control queries, and allows the system to synchronize itself to a system whose address passes the access list criteria.
4. serve —Allows time requests and NTP control queries, but does not allow the system to synchronize itself to a system whose address passes the access list criteria.

5. serve-only —Allows only time requests from a system whose address passes the access list criteria.
6. query-only —Allows only NTP control queries from a system whose address passes the access list criteria.

If the source IP address matches the access lists for more than one access type, the first type is granted access. If no access groups are specified, all access types are granted access to all systems. If any access groups are specified, only the specified access types will be granted access.

For details on NTP control queries, see RFC 1305.

The encrypted NTP authentication scheme should be used when a reliable form of access control is required. Unlike the access list-based restriction scheme that is based on IP addresses, the encrypted authentication scheme uses authentication keys and an authentication process to determine if NTP synchronization packets sent by designated peers or servers on a local network are deemed as trusted before the time information that they carry along with them is accepted.

The authentication process begins from the moment an NTP packet is created. Cryptographic checksum keys are generated using the message digest algorithm 5 (MD5) and are embedded into the NTP synchronization packet that is sent to a receiving client. Once a packet is received by a client, its cryptographic checksum key is decrypted and checked against a list of trusted keys. If the packet contains a matching authentication key, the time-stamp information that is contained within the packet is accepted by the receiving client. NTP synchronization packets that do not contain a matching authenticator key are ignored.

Note In large networks, where many trusted keys must be configured, the Range of Trusted Key Configuration feature enables configuring multiple keys simultaneously.

It is important to note that the encryption and decryption processes used in NTP authentication can be very CPU-intensive and can seriously degrade the accuracy of the time that is propagated within a network. If your network setup permits a more comprehensive model of access control, you should consider the use of the access list-based form of control.

After NTP authentication is properly configured, your networking device will synchronize with and provide synchronization only to trusted time sources.

NTP Services on a Specific Interface

Network Time Protocol (NTP) services are disabled on all interfaces by default. NTP is enabled globally when any NTP commands are entered. You can selectively prevent NTP packets from being received through a specific interface by using the **ntp disable** command in interface configuration mode.

Source IP Address for NTP Packets

When the system sends an NTP packet, the source IP address is normally set to the address of the interface through which the NTP packet is sent. Use the **ntp source** *interface* command in global configuration mode to configure a specific interface from which the IP source address will be taken.

This interface will be used for the source address for all packets sent to all destinations. If a source address is to be used for a specific association, use the **source** keyword in the **ntp peer** or **ntp server** command.

NTP Implementation

Implementation of NTP does not support stratum 1 service; it is not possible to connect to a radio or atomic clock. We recommend that the time service for your network be derived from the public NTP servers available on the IP Internet.

If the network is isolated from the Internet, NTP allows a device to act as if it is synchronized through NTP, when in fact it has learned the time by using other means. Other devices then synchronize to that device through NTP.

When multiple sources of time are available, NTP is always considered to be more authoritative. NTP time overrides the time set by any other method.

Several manufacturers include NTP software for their host systems, and a publicly available version for systems running UNIX and its various derivatives is also available. This software allows host systems to be time-synchronized as well.

DNS

The DNS protocol controls the Domain Name System (DNS), a distributed database with which you can map hostnames to IP addresses. When you configure DNS on your device, you can substitute the hostname for the IP address with all IP commands, such as **ping**, **telnet**, **connect**, and related Telnet support operations.

IP defines a hierarchical naming scheme that allows a device to be identified by its location or domain. Domain names are pieced together with periods (.) as the delimiting characters. For example, Cisco Systems is a commercial organization that IP identifies by a *com* domain name, so its domain name is *cisco.com*. A specific device in this domain, for example, the File Transfer Protocol (FTP) system is identified as *ftp.cisco.com*.

To keep track of domain names, IP has defined the concept of a domain name server, which holds a cache (or database) of names mapped to IP addresses. To map domain names to IP addresses, you must first identify the hostnames, specify the name server that is present on your network, and enable the DNS.

Default DNS Settings

Table 139: Default DNS Settings

Feature	Default Setting
DNS enable state	Enabled.
DNS default domain name	None configured.
DNS servers	No name server addresses are configured.

Login Banners

You can configure a message-of-the-day (MOTD) and a login banner. The MOTD banner is displayed on all connected terminals at login and is useful for sending messages that affect all network users (such as impending system shutdowns).

The login banner is also displayed on all connected terminals. It appears after the MOTD banner and before the login prompts.

In default banner configuration, the MOTD and login banners are not configured

Note For complete syntax and usage information for the commands used in this section, see the *Cisco IOS Configuration Fundamentals Command Reference, Release 12.4*.

Default Banner Configuration

The MOTD and login banners are not configured.

MAC Address Table

The MAC address table contains address information that the device uses to forward traffic between ports. All MAC addresses in the address table are associated with one or more ports. The address table includes these types of addresses:

- Dynamic address—A source MAC address that the device learns and then ages when it is not in use.
- Static address—A manually entered unicast address that does not age and that is not lost when the device resets.

The address table lists the destination MAC address, the associated VLAN ID, and port number associated with the address and the type (static or dynamic).

Note For complete syntax and usage information for the commands used in this section, see the command reference for this release.

MAC Address Table Creation

With multiple MAC addresses supported on all ports, you can connect any port on the device to other network devices. The device provides dynamic addressing by learning the source address of packets it receives on each port and adding the address and its associated port number to the address table. As devices are added or removed from the network, the device updates the address table, adding new dynamic addresses and aging out those that are not in use.

The aging interval is globally configured. However, the device maintains an address table for each VLAN, and STP can accelerate the aging interval on a per-VLAN basis.

The device sends packets between any combination of ports, based on the destination address of the received packet. Using the MAC address table, the device forwards the packet only to the port associated with the destination address. If the destination address is on the port that sent the packet, the packet is filtered and not forwarded. The device always uses the store-and-forward method: complete packets are stored and checked for errors before transmission.

MAC Addresses and VLANs

All addresses are associated with a VLAN. An address can exist in more than one VLAN and have different destinations in each. Unicast addresses, for example, could be forwarded to port 1 in VLAN 1 and ports 9, 10, and 1 in VLAN 5.

Each VLAN maintains its own logical address table. A known address in one VLAN is unknown in another until it is learned or statically associated with a port in the other VLAN.

Default MAC Address Table Settings

The following table shows the default settings for the MAC address table.

Table 140: Default Settings for the MAC Address

Feature	Default Setting
Aging time	300 seconds
Dynamic addresses	Automatically learned
Static addresses	None configured

ARP Table Management

To communicate with a device (over Ethernet, for example), the software first must learn the 48-bit MAC address or the local data link address of that device. The process of learning the local data link address from an IP address is called *address resolution*.

The Address Resolution Protocol (ARP) associates a host IP address with the corresponding media or MAC addresses and the VLAN ID. Using an IP address, ARP finds the associated MAC address. When a MAC address is found, the IP-MAC address association is stored in an ARP cache for rapid retrieval. Then the IP datagram is encapsulated in a link-layer frame and sent over the network. Encapsulation of IP datagrams and ARP requests and replies on IEEE 802 networks other than Ethernet is specified by the Subnetwork Access Protocol (SNAP). By default, standard Ethernet-style ARP encapsulation (represented by the **arpa** keyword) is enabled on the IP interface.

ARP entries added manually to the table do not age and must be manually removed.

For CLI procedures, see the Cisco IOS Release 12.4 documentation on *Cisco.com*.

How to Administer the Device

Configuring the Time and Date Manually

System time remains accurate through restarts and reboot, however, you can manually configure the time and date after the system is restarted.

We recommend that you use manual configuration only when necessary. If you have an outside source to which the device can synchronize, you do not need to manually set the system clock.

Setting the System Clock

If you have an outside source on the network that provides time services, such as an NTP server, you do not need to manually set the system clock.

Follow these steps to set the system clock:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Use one of the following: • **clock set** *hh:mm:ss day month year* • **clock set** *hh:mm:ss month day year* **Example:** `Device# clock set 13:32:00 23 March 2013`	Manually set the system clock using one of these formats: • *hh:mm:ss*—Specifies the time in hours (24-hour format), minutes, and seconds. The time specified is relative to the configured time zone. • *day*—Specifies the day by date in the month. • *month*—Specifies the month by name. • *year*—Specifies the year (no abbreviation).

Configuring the Time Zone

Follow these steps to manually configure the time zone:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**clock timezone** *zone hours-offset* [*minutes-offset*] **Example:** `Device(config)# clock timezone AST -3 30`	Sets the time zone. Internal time is kept in Coordinated Universal Time (UTC), so this command is used only for display purposes and when the time is manually set.

	Command or Action	Purpose
		• *zone*—Enters the name of the time zone to be displayed when standard time is in effect. The default is UTC. • *hours-offset*—Enters the hours offset from UTC. • (Optional) *minutes-offset*—Enters the minutes offset from UTC. This available where the local time zone is a percentage of an hour different from UTC.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring NTP

These following sections provide configuration information on NTP:

Default NTP Configuration

shows the default NTP configuration.

Table 141: Default NTP Configuration

Feature	Default Setting
NTP authentication	Disabled. No authentication key is specified.
NTP peer or server associations	None configured.
NTP broadcast service	Disabled; no interface sends or receives NTP broadcast packets.

Feature	Default Setting
NTP access restrictions	No access control is specified.
NTP packet source IP address	The source address is set by the outgoing interface.

NTP is enabled on all interfaces by default. All interfaces receive NTP packets.

Configuring NTP Authentication

To configure NTP authentication, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	[**no**] **ntp authenticate** **Example:** `Device(config)# ntp authenticate`	Enables NTP authentication. Use the **no** form of this command to disable NTP authentication
Step 4	[**no**] **ntp authentication-key** *number* {**md5** | **cmac-aes-128** | **hmac-sha1** | **hmac-sha2-256**} *value* **Example:** `Device(config)# ntp authentication-key 42 md5 aNiceKey`	Defines the authentication keys. • Each key has a key number, a type, and a value. • Keys can be one of the following types: • **md5**: Authentication using the MD5 algorithm. • **cmac-aes-128**: Authentication using Cipher-based message authentication codes (CMAC) with the AES-128 algorithm. The digest length is 128 bits and the key length is 16 or 32 bytes. • **hmac-sha1**: Authentication using Hash-based Message Authentication

	Command or Action	Purpose
		Code (HMAC) using the SHA1 hash function. The digest length is 128 bits and the key length is 1 to 32 bytes. • **hmac-sha2-256**: Authentication using HMAC using the SHA2 hash function. The digest length is 256 bits and the key length is 1 to 32 bytes Use the **no** form of this command to remove authentication key.
Step 5	[**no**] **ntp trusted-key** *key-number* **Example:** `Device(config)# ntp trusted-key 42`	Defines trusted authentication keys that a peer NTP device must provide in its NTP packets for this device to synchronize to it. Use the **no** form of this command to disable trusted authentication.
Step 6	[**no**] **ntp server** *ip-address* **key** *key-id* [**prefer**] **Example:** `Device(config)# ntp server 172.16.22.44 key 42`	Allows the software clock to be synchronized by an NTP time server. • *ip-address*: The IP address of the time server providing the clock synchronization. • *key-id*: Authentication key defined with the **ntp authentication-key** command. • **prefer**: Sets this peer as the preferred one that provides synchronization. This keyword reduces clock hop among peers. Use the **no** form of this command to remove a server association.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring Poll-Based NTP Associations

To configure poll-based NTP associations, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	[**no**] **ntp peer** *ip-address* [**version** *number*] [**key** *key-id*] [**source** *interface*] [**prefer**] **Example:** `Device(config)# ntp peer 172.16.22.44 version 2`	Configures the device system clock to synchronize a peer or to be synchronized by a peer (peer association). • *ip-address*: The IP address of the peer providing or being provided, the clock synchronization. • *number*: NTP version number. The range is 1 to 4. By default, version 4 is selected. • *key-id*: Authentication key defined with the **ntp authentication-key** command. • *interface*: The interface from which to pick the IP source address. By default, the source IP address is taken from the outgoing interface. • **prefer**: Sets this peer as the preferred one that provides synchronization. This keyword reduces switching back and forth between peers. Use the **no** form of this command to remove a peer association.
Step 4	[**no**] **ntp server** [**vrf** *vrf-name*] *ip-address* [**version** *number*] [**key** *key-id*] [**source** *interface*] [**prefer**] **Example:** `Device(config)# ntp server 172.16.22.44 version 2`	Configures the device's system clock to be synchronized by a time server (server association). • *vrf-name*: The virtual routing and forwarding (VRF) address of the server providing the clock synchronization. **Note** Before you configure this command, the VRF must be configured. • *ip-address*: The IP address of the time server providing the clock synchronization. • *number*: NTP version number. The range is 1 to 4. By default, version 4 is selected.

	Command or Action	Purpose
		• *key-id*: Authentication key defined with the **ntp authentication-key** command. • *interface*: The interface from which to pick the IP source address. By default, the source IP address is taken from the outgoing interface. • **prefer**: Sets this peer as the preferred one that provides synchronization. This keyword reduces clock hop among peers. Use the **no** form of this command to remove a server association.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring Broadcast-Based NTP Associations

To configure broadcast-based NTP associations, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Configures an interface and enters interface configuration mode.
Step 4	[**no**] **ntp broadcast** [**version** *number*] [**key** *key-id*] [*destination-address*]	Enables the interface to send NTP broadcast packets to a peer.

	Command or Action	Purpose
	Example: `Device(config-if)# ntp broadcast version 2`	• *number*: NTP version number. The range is 1 to 4. By default, version 4 is used. • *key-id*: Authentication key. • *destination-address*: IP address of the peer that is synchronizing its clock to this switch. Use the **no** form of this command to disable the interface from sending NTP broadcast packets.
Step 5	[**no**] **ntp broadcast client** **Example:** `Device(config-if)# ntp broadcast client`	Enables the interface to receive NTP broadcast packets. Use the **no** form of this command to disable the interface from receiving NTP broadcast packets.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Returns to privileged EXEC mode.
Step 7	[**no**] **ntp broadcastdelay** *microseconds* **Example:** `Device(config)# ntp broadcastdelay 100`	(Optional) Change the estimated round-trip delay between the device and the NTP broadcast server The default is 3000 microseconds. The range is from 1 to 999999. Use the **no** form of this command to disable the interface from receiving NTP broadcast packets.
Step 8	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring NTP Access Restrictions

You can control NTP access on two levels as described in these sections:

Creating an Access Group and Assigning a Basic IP Access List

To create an access group and assign a basic IP access list, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	[**no**] **ntp access-group** {**query-only** | **serve-only** | **serve** | **peer**} *access-list-number* **Example:** `Device(config)# ntp access-group peer 99`	Create an access group, and apply a basic IP access list.. • **query-only**: NTP control queries. • **serve-only**: Time requests. • **serve**: Allows time requests and NTP control queries, but does not allow the device to synchronize to the remote device. • **peer**: Allows time requests and NTP control queries and allows the device to synchronize to the remote device. • *access-list-number*: IP access list number. The range is from 1 to 99. Use the **no** form of this command to remove access control to the switch NTP services.
Step 4	**access-list** *access-list-number* **permit** *source* [*source-wildcard*] **Example:** `Device(config)# access-list 99 permit 172.20.130.5`	Create the access list. • *access-list-number*: IP access list number. The range is from 1 to 99. • **permit**: Permits access if the conditions are matched. • *source*: IP address of the device that is permitted access to the device. • *source-wildcard*: Wildcard bits to be applied to the source. **Note** When creating an access list, remember that, by default, the end of the access list contains

	Command or Action	Purpose
		an implicit deny statement for everything if it did not find a match before reaching the end. Use the **no** form of this command to remove authentication key.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Disabling NTP Services on a Specific Interface

To disable NTP packets from being received on an interface, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Enters global configuration mode.
Step 4	[**no**] **ntp disable** **Example:** `Device(config-if)# ntp disable`	Disables NTP packets from being received on the interface. Use the **no** form of this command to re-enable receipt of NTP packets on an interface.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	

Configuring a System Name

Follow these steps to manually configure a system name:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**hostname** *name* **Example:** `Device(config)# hostname remote-users`	Configures a system name. When you set the system name, it is also used as the system prompt. The default setting is Switch. The name must follow the rules for ARPANET hostnames. They must start with a letter, end with a letter or digit, and have as interior characters only letters, digits, and hyphens. Names can be up to 63 characters.
Step 4	**end** **Example:** `remote-users(config)#end` `remote-users#`	Returns to priviliged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:**	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`Device# copy running-config startup-config`	

Setting Up DNS

If you use the device IP address as its hostname, the IP address is used and no DNS query occurs. If you configure a hostname that contains no periods (.), a period followed by the default domain name is appended to the hostname before the DNS query is made to map the name to an IP address. The default domain name is the value set by the **ip domain name** command in global configuration mode. If there is a period (.) in the hostname, the Cisco IOS software looks up the IP address without appending any default domain name to the hostname.

Follow these steps to set up your switch to use the DNS:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip domain name** *name* **Example:** `Device(config)# ip domain name Cisco.com`	Defines a default domain name that the software uses to complete unqualified hostnames (names without a dotted-decimal domain name). Do not include the initial period that separates an unqualified name from the domain name. At boot time, no domain name is configured; however, if the device configuration comes from a BOOTP or Dynamic Host Configuration Protocol (DHCP) server, then the default domain name might be set by the BOOTP or DHCP server (if the servers were configured with this information).
Step 4	**ip name-server** *server-address1* [*server-address2 ... server-address6*] **Example:**	Specifies the address of one or more name servers to use for name and address resolution.

	Command or Action	Purpose
	`Device(config)# ip name-server 192.168.1.100 192.168.1.200 192.168.1.300`	You can specify up to six name servers. Separate each server address with a space. The first server specified is the primary server. The device sends DNS queries to the primary server first. If that query fails, the backup servers are queried.
Step 5	**ip domain lookup** [**nsap** | **source-interface** *interface*] **Example:** `Device(config)# ip domain-lookup`	(Optional) Enables DNS-based hostname-to-address translation on your device. This feature is enabled by default. If your network devices require connectivity with devices in networks for which you do not control name assignment, you can dynamically assign device names that uniquely identify your devices by using the global Internet naming scheme (DNS).
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Message-of-the-Day Login Banner

You can create a single or multiline message banner that appears on the screen when someone logs in to the device.

Follow these steps to configure a MOTD login banner:

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**banner motd** *c message c* **Example:** `Device(config)# banner motd #` `This is a secure site. Only` `authorized users are allowed.` `For access, contact technical` `support.` `#`	Specifies the message of the day. *c*—Enters the delimiting character of your choice, for example, a pound sign (#), and press the **Return** key. The delimiting character signifies the beginning and end of the banner text. Characters after the ending delimiter are discarded. *message*—Enters a banner message up to 255 characters. You cannot use the delimiting character in the message.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Login Banner

You can configure a login banner to be displayed on all connected terminals. This banner appears after the MOTD banner and before the login prompt.

Follow these steps to configure a login banner:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**banner login** *c message c* **Example:** `Device(config)# banner login $` `Access for authorized users only.` `Please enter your username and password.` `$`	Specifies the login message. *c*— Enters the delimiting character of your choice, for example, a pound sign (#), and press the **Return** key. The delimiting character signifies the beginning and end of the banner text. Characters after the ending delimiter are discarded. *message*—Enters a login message up to 255 characters. You cannot use the delimiting character in the message.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Managing the MAC Address Table

Changing the Address Aging Time

Follow these steps to configure the dynamic address table aging time:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mac address-table aging-time** [*0* | *10-1000000*] [**routed-mac** | **vlan** *vlan-id*] **Example:** `Device(config)# mac address-table aging-time 500 vlan 2`	Sets the length of time that a dynamic entry remains in the MAC address table after the entry is used or updated. The range is 10 to 1000000 seconds. The default is 300. You can also enter 0, which disables aging. Static address entries are never aged or removed from the table. *vlan-id*—Valid IDs are 1 to 4094.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring MAC Address Change Notification Traps

Follow these steps to configure the switch to send MAC address change notification traps to an NMS host:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server host** *host-addr community-string notification-type* { **informs** | **traps** } {**version** {**1** | **2c** | **3**}} {**vrf** *vrf instance name*} **Example:** `Device(config)# snmp-server host 172.20.10.10 traps private mac-notification`	Specifies the recipient of the trap message. • *host-addr*—Specifies the name or address of the NMS. • **traps** (the default)—Sends SNMP traps to the host. • **informs**—Sends SNMP informs to the host. • **version**—Specifies the SNMP version to support. Version 1, the default, is not available with informs. • *community-string*—Specifies the string to send with the notification operation. Though you can set this string by using the **snmp-server host** command, we recommend that you define this string by using the **snmp-server community** command before using the **snmp-server host** command. • *notification-type*—Uses the **mac-notification** keyword. • **vrf** *vrf instance name*—Specifies the VPN routing/forwarding instance for this host.
Step 4	**snmp-server enable traps mac-notification change** **Example:**	Enables the device to send MAC address change notification traps to the NMS.

	Command or Action	Purpose
	`Device(config)# snmp-server enable traps mac-notification change`	
Step 5	**mac address-table notification change** **Example:** `Device(config)# mac address-table notification change`	Enables the MAC address change notification feature.
Step 6	**mac address-table notification change** [**interval** *value*] [**history-size** *value*] **Example:** `Device(config)# mac address-table notification change interval 123` `Device(config)#mac address-table notification change history-size 100`	Enters the trap interval time and the history table size. • (Optional) **interval** *value*—Specifies the notification trap interval in seconds between each set of traps that are generated to the NMS. The range is 0 to 2147483647 seconds; the default is 1 second. • (Optional) **history-size** *value*—Specifies the maximum number of entries in the MAC notification history table. The range is 0 to 500; the default is 1.
Step 7	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Enters interface configuration mode, and specifies the Layer 2 interface on which to enable the SNMP MAC address notification trap.
Step 8	**snmp trap mac-notification change** {**added** | **removed**} **Example:** `Device(config-if)# snmp trap mac-notification change added`	Enables the MAC address change notification trap on the interface. • Enables the trap when a MAC address is **added** on this interface. • Enables the trap when a MAC address is **removed** from this interface.
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 10	**show running-config** **Example:**	Verifies your entries.

	Command or Action	Purpose
	`Device# show running-config`	
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring MAC Address Move Notification Traps

When you configure MAC-move notification, an SNMP notification is generated and sent to the network management system whenever a MAC address moves from one port to another within the same VLAN.

Follow these steps to configure the device to send MAC address-move notification traps to an NMS host:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server host** *host-addr* {**traps** | **informs**} {**version** {**1** | **2c** | **3**}} *community-string notification-type* **Example:** `Device(config)# snmp-server host 172.20.10.10 traps private mac-notification`	Specifies the recipient of the trap message. • *host-addr*—Specifies the name or address of the NMS. • **traps** (the default)—Sends SNMP traps to the host. • **informs**—Sends SNMP informs to the host. • **version**—Specifies the SNMP version to support. Version 1, the default, is not available with informs. • *community-string*—Specifies the string to send with the notification operation. Though you can set this string by using

	Command or Action	Purpose
		the **snmp-server host** command, we recommend that you define this string by using the **snmp-server community** command before using the **snmp-server host** command. • *notification-type*—Uses the **mac-notification** keyword.
Step 4	**snmp-server enable traps mac-notification move** **Example:** `Device(config)# snmp-server enable traps mac-notification move`	Enables the device to send MAC address move notification traps to the NMS.
Step 5	**mac address-table notification mac-move** **Example:** `Device(config)# mac address-table notification mac-move`	Enables the MAC address move notification feature.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

To disable MAC address-move notification traps, use the **no snmp-server enable traps mac-notification move** global configuration command. To disable the MAC address-move notification feature, use the **no mac address-table notification mac-move** global configuration command.

You can verify your settings by entering the **show mac address-table notification mac-move** privileged EXEC commands.

Configuring MAC Threshold Notification Traps

When you configure MAC threshold notification, an SNMP notification is generated and sent to the network management system when a MAC address table threshold limit is reached or exceeded.

Follow these steps to configure the switch to send MAC address table threshold notification traps to an NMS host:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server host** *host-addr* {**traps** | **informs**} {**version** {**1** | **2c** | **3**}} *community-string notification-type* **Example:** `Device(config)# snmp-server host 172.20.10.10 traps private mac-notification`	Specifies the recipient of the trap message. • *host-addr*—Specifies the name or address of the NMS. • **traps** (the default)—Sends SNMP traps to the host. • **informs**—Sends SNMP informs to the host. • **version**—Specifies the SNMP version to support. Version 1, the default, is not available with informs. • *community-string*—Specifies the string to send with the notification operation. You can set this string by using the **snmp-server host** command, but we recommend that you define this string by using the **snmp-server community** command before using the **snmp-server host** command. • *notification-type*—Uses the **mac-notification** keyword.

	Command or Action	Purpose
Step 4	**snmp-server enable traps mac-notification threshold** **Example:** `Device(config)# snmp-server enable traps mac-notification threshold`	Enables MAC threshold notification traps to the NMS.
Step 5	**mac address-table notification threshold** **Example:** `Device(config)# mac address-table notification threshold`	Enables the MAC address threshold notification feature.
Step 6	**mac address-table notification threshold** [**limit** *percentage*] | [**interval** *time*] **Example:** `Device(config)# mac address-table notification threshold interval 123` `Device(config)# mac address-table notification threshold limit 78`	Enters the threshold value for the MAC address threshold usage monitoring. • (Optional) **limit** *percentage*—Specifies the percentage of the MAC address table use; valid values are from 1 to 100 percent. The default is 50 percent. • (Optional) **interval** *time*—Specifies the time between notifications; valid values are greater than or equal to 120 seconds. The default is 120 seconds.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Disabling MAC Address Learning on VLAN

You can control MAC address learning on a VLAN to manage the available MAC address table space by controlling which VLANs can learn MAC addresses. Before you disable MAC address learning, be sure that you are familiar with the network topology. Disabling MAC address learning on VLAN could cause flooding in the network.

Beginning in privileged EXEC mode, follow these steps to disable MAC address learning on a VLAN:

Before you begin

Follow these guidelines when disabling MAC address learning on a VLAN:

- Use caution before disabling MAC address learning on a VLAN with a configured switch virtual interface (SVI). The switch then floods all IP packets in the Layer 2 domain.
- You can disable MAC address learning on a single VLAN ID from 2 - 4094 (for example, no mac address-table learning vlan 223) or a range of VLAN IDs, separated by a hyphen or comma (for example, no mac address-table learning vlan 1-10, 15).
- It is recommended that you disable MAC address learning only in VLANs with two ports. If you disable MAC address learning on a VLAN with more than two ports, every packet entering the switch is flooded in that VLAN domain.
- If you disable MAC address learning on a VLAN that includes a secure port, MAC address learning is not disabled on that port.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters the global configuration mode.
Step 2	**no mac-address-table learning vlan**[*vlan-id* |,*vlan-id*	*-vlan-id*,] **Example:** `Device(config)# no mac-address-table learning {vlan vlan-id [,vlan-id	-vlan-id]`	Disable MAC address learning on a specified VLAN or VLANs. You can specify a single VLAN ID or a range of VLAN IDs separated by a hyphen or comma. Valid VLAN IDs range from 2 - 4094. It cannot be an internal VLAN.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 4	**show mac-address-table learning vlan**[*vlan-id*] **Example:** `Device# show mac-address-table learning [vlan vlan-id]`	Verify the configuration. You can display the MAC address learning status of all VLANs or a specified VLAN by entering the show mac-address-table learning [vlan vlan-id] privileged EXEC command.

	Command or Action	Purpose
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Save your entries in the configuration file.
Step 6	**default mac address-table learning** **Example:** `Device# default mac address-table`	(Optional) Reenable MAC address learning on VLAN in a global configuration mode.

Adding and Removing Static Address Entries

Follow these steps to add a static address:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mac address-table static** *mac-addr* **vlan** *vlan-id* **interface** *interface-id* **Example:** `Device(config)# mac address-table static c2f3.220a.12f4 vlan 4 interface gigabitethernet 1/1`	Adds a static address to the MAC address table. • *mac-addr*—Specifies the destination MAC unicast address to add to the address table. Packets with this destination address received in the specified VLAN are forwarded to the specified interface. • *vlan-id*—Specifies the VLAN for which the packet with the specified MAC address is received. Valid VLAN IDs are 1 to 4094. • *interface-id*—Specifies the interface to which the received packet is forwarded. Valid interfaces include physical ports or port channels. For static multicast addresses, you can enter multiple interface IDs. For static unicast addresses, you can enter only one interface at a time, but you

	Command or Action	Purpose
		can enter the command multiple times with the same MAC address and VLAN ID.
Step 4	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Unicast MAC Address Filtering

Follow these steps to configure the device to drop a source or destination unicast static address:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mac address-table static** *mac-addr* **vlan** *vlan-id* **drop** **Example:** `Device(config)# mac address-table static c2f3.220a.12f4 vlan 4 drop`	Enables unicast MAC address filtering and configure the device to drop a packet with the specified source or destination unicast static address. • *mac-addr*—Specifies a source or destination unicast MAC address (48-bit). Packets with this MAC address are dropped. • *vlan-id*—Specifies the VLAN for which the packet with the specified MAC address is received. Valid VLAN IDs are 1 to 4094.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring and Maintaining Administration of the Device

Command	Purpose
clear mac address-table dynamic	Removes all dynamic entries.
clear mac address-table dynamic address *mac-address*	Removes a specific MAC address.
clear mac address-table dynamic interface *interface-id*	Removes all addresses on the specified physical port or port channel.
clear mac address-table dynamic vlan *vlan-id*	Removes all addresses on a specified VLAN.
show clock [*detail*]	Displays the time and date configuration.
show ip igmp snooping groups	Displays the Layer 2 multicast entries for all VLANs or the specified VLAN.
show mac address-table address *mac-address*	Displays MAC address table information for the specified MAC address.
show mac address-table aging-time	Displays the aging time in all VLANs or the specified VLAN.
show mac address-table count	Displays the number of addresses present in all VLANs or the specified VLAN.
show mac address-table dynamic	Displays only dynamic MAC address table entries.
show mac address-table interface *interface-name*	Displays the MAC address table information for the specified interface.
show mac address-table move update	Displays the MAC address table move update information.
show mac address-table multicast	Displays a list of multicast MAC addresses.

Command	Purpose
show mac address-table notification {change | mac-move | threshold}	Displays the MAC notification parameters and history table.
show mac address-table secure	Displays the secure MAC addresses.
show mac address-table static	Displays only static MAC address table entries.
show mac address-table vlan *vlan-id*	Displays the MAC address table information for the specified VLAN.

Configuration Examples for Device Administration

Example: Setting the System Clock

This example shows how to manually set the system clock:

```
Device# clock set 13:32:00 23 July 2013
```

Examples: Configuring Summer Time

This example (for daylight savings time) shows how to specify that summer time starts on March 10 at 02:00 and ends on November 3 at 02:00:

```
Device(config)# clock summer-time PDT recurring PST date
10 March 2013 2:00 3 November 2013 2:00
```

This example shows how to set summer time start and end dates:

```
Device(config)#clock summer-time PST date
20 March 2013 2:00 20 November 2013 2:00
```

Example: Configuring a MOTD Banner

This example shows how to configure a MOTD banner by using the pound sign (#) symbol as the beginning and ending delimiter:

```
Device(config)# banner motd #

This is a secure site. Only authorized users are allowed.
For access, contact technical support.

#

Device(config)#
```

This example shows the banner that appears from the previous configuration:

```
Unix> telnet 192.0.2.15

Trying 192.0.2.15...

Connected to 192.0.2.15.

Escape character is '^]'.

This is a secure site. Only authorized users are allowed.

For access, contact technical support.

User Access Verification

Password:
```

Example: Configuring a Login Banner

This example shows how to configure a login banner by using the dollar sign ($) symbol as the beginning and ending delimiter:

```
Device(config)# banner login $

Access for authorized users only. Please enter your username and password.

$

Device(config)#
```

Example: Configuring MAC Address Change Notification Traps

This example shows how to specify 172.20.10.10 as the NMS, enable MAC address notification traps to the NMS, enable the MAC address-change notification feature, set the interval time to 123 seconds, set the history-size to 100 entries, and enable traps whenever a MAC address is added on the specified port:

```
Device(config)# snmp-server host 172.20.10.10 traps private mac-notification
Device(config)# snmp-server enable traps mac-notification change
Device(config)# mac address-table notification change
Device(config)# mac address-table notification change interval 123
Device(config)# mac address-table notification change history-size 100
Device(config)# interface gigabitethernet1/1
Device(config-if)# snmp trap mac-notification change added
```

Example: Configuring MAC Threshold Notification Traps

This example shows how to specify 172.20.10.10 as the NMS, enable the MAC address threshold notification feature, set the interval time to 123 seconds, and set the limit to 78 per cent:

```
Device(config)# snmp-server host 172.20.10.10 traps private mac-notification
```

```
Device(config)# snmp-server enable traps mac-notification threshold
Device(config)# mac address-table notification threshold
Device(config)# mac address-table notification threshold interval 123
Device(config)# mac address-table notification threshold limit 78
```

Example: Adding the Static Address to the MAC Address Table

This example shows how to add the static address c2f3.220a.12f4 to the MAC address table. When a packet is received in VLAN 4 with this MAC address as its destination address, the packet is forwarded to the specified port:

Note You cannot associate the same static MAC address to multiple interfaces. If the command is executed again with a different interface, the static MAC address is overwritten on the new interface.

```
Device(config)# mac address-table static c2f3.220a.12f4 vlan 4 interface gigabitethernet1/1
```

Example: Configuring Unicast MAC Address Filtering

This example shows how to enable unicast MAC address filtering and how to configure drop packets that have a source or destination address of c2f3.220a.12f4. When a packet is received in VLAN 4 with this MAC address as its source or destination, the packet is dropped:

```
Device(config)# mac address-table static c2f3.220a.12f4 vlan 4 drop
```

CHAPTER 135

Boot Integrity Visibility

Information About Boot Integrity Visibility

Boot Integrity Visibility allows Cisco's platform identity and software integrity information to be visible and actionable. Platform identity provides the platform's manufacturing installed identity. Software integrity exposes boot integrity measurements that can be used to assess whether the platform has booted trusted code.

During the boot process, the software creates a checksum record of each stage of the bootloader activities.

You can retrieve this record and compare it with a Cisco-certified record to verify if your software image is genuine. If the checksum values do not match, you may be running a software image that is either not certified by Cisco or has been altered by an unauthorized party.

Image Signing and Bootup

The Cisco build servers generate the Cisco IOS XE images. Cisco IOS XE images use the Abraxas image signing system to sign these images securely with the Cisco private RSA keys.

When you copy the Cisco IOS XE image onto a Switch, Cisco's ROMMON Boot ROM verifies the image using Cisco release keys. These keys are public keys that correspond to the Cisco release private key that is stored securely on the Abraxas servers. The release key is stored in the ROMMON.

Switches support boot integrity visibility feature. Boot integrity visibility serves as a hardware trust anchor which validates the ROMMON software to ensure that the ROMMON software is not tampered with.

The Cisco IOS XE image is digitally signed during the build time. An SHA-512 hash is generated over the entire binary image file, and then the hash is encrypted with a Cisco RSA 2048-bit private key. The ROMMON verifies the signature using the Cisco public key. If the software is not generated by a Cisco build system, the signature verification fails. The device ROMMON rejects the image and stops booting. If the signature verification is successfully, the device boots the image to the Cisco IOS XE runtime environment.

The ROMMON follows these steps when it verifies a signed Cisco IOS XE image during the bootup:

1. Loads the Cisco IOS XE image into the CPU memory.

2. Examines the Cisco IOS XE package header.
3. Runs a non-secure integrity check on the image to ensure that there is no unintentional file corruption from the disk or TFTP. This is performed using a non-secure SHA-1 hash.
4. Copies the Cisco's RSA 2048-bit public release key from the ROMMON storage and validates that the Cisco's RSA 2048-bit public release key is not tampered.
5. Extracts the Code Signing signature (SHA-512 hash) from the package header and verifies it using Cisco's RSA 2048-bit public release key.
6. Performs the Code Signing validation by calculating the SHA-512 hash of the Cisco IOS XE package and compares it with the Code Signing signature. The Signed package is now validated.
7. Examines the Cisco IOS XE package header to validate the platform type and CPU architecture for compatibility.
8. Extracts the Cisco IOS XE software from the Cisco IOS XE package and boots it.

Note In above process, step 3 is a non-secure check of the image which is intended to confirm the image against inadvertent corruption due to disk errors, file transfer errors, or copying errors. This is not part of the image code signing. This check is not intended to detect deliberate image tampering.

Image Code Signing validation occurs in step 4, 5, and 6. This is a secure code signing check of the image using an SHA-512 hash that is encrypted with a 2048-bit RSA key. This check is intended to detect deliberate image tampering.

Verifying the Software Image and Hardware

This task describes how to retrieve the checksum record that was created during a switch bootup. Enter the following commands in privileged EXEC mode.

Note On executing the following commands, you might see the message **% Please Try After Few Seconds** displayed on the CLI. This does not indicate a CLI failure, but indicates setting up of underlying infrastructure required to get the required output. We recommend waiting for a few minutes and then try the command again.

The messages **% Error retrieving SUDI certificate** and **% Error retrieving integrity data** signify a real CLI failure.

Procedure

	Command or Action	Purpose
Step 1	**show platform sudi certificate** [**sign** [**nonce** *nonce*]] **Example:**	Displays checksum record for the specific SUDI. • (Optional) **sign** - Show signature

	Command or Action	Purpose
	`Device# show platform sudi certificate sign nonce 123`	• (Optional) **nonce** - Enter a nonce value
Step 2	**show platform integrity** [**sign** [**nonce** *nonce*]] **Example:** `Device# show platform integrity sign nonce 123`	Displays checksum record for boot stages. • (Optional) **sign** - Show signature • (Optional) **nonce** - Enter a nonce value

Verifying Platform Identity and Software Integrity

Verifying Platform Identity

The following example displays the Secure Unique Device Identity (SUDI) chain in PEM format. Encoded into the SUDI is the Product ID and Serial Number of each individual device such that the device can be uniquely identified on a network of thousands of devices. The first certificate is the Cisco Root CA 2048 and the second is the Cisco subordinate CA (ACT2 SUDI CA). Both certificates can be verified to match those published on https://www.cisco.com/security/pki/. The third is the SUDI certificate.

```
Device# show platform sudi certificate sign nonce 123
-----BEGIN CERTIFICATE-----
MIIDQzCCAiugAwIBAgIQX/h7KCtU3I1CoxW1aMmt/zANBgkqhkiG9w0BAQUFADA1
MRYwFAYDVQQKEw1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENB
IDIwNDgwHhcNMDQwNTE0MjAxNzEyWhcNMjkwNTE0MjAyNTQyWjA1MRYwFAYDVQQK
Ew1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwNDgwggEg
MA0GCSqGSIb3DQEBAQUAA4IBDQAwggEIAoIBAQCwmrmrp68Kd6ficba0ZmKUeIhH
xmJVhEAyv8CrLqUccda8bnuoqrpu0hWISEWdovyD0My5jOAmaHBKeN8hF570YQXJ
FcjPFto1YYmUQ6iEqDGYeJu5Tm8sUxJszR2tKyS7McQr/4NEb7Y9JHcJ6r8qqB9q
VvYgDxFUl4F1pyXOWWqCZe+36ufijXWLbvLdT6ZeYpzPEApk0E5tzivMW/VgpSdH
jWn0f84bcN5wGyDWbs2mAag8EtKpP6BrXruOIIt6keO1aO6g58QBdKhTCytKmg9l
Eg6CTY5j/e/rmxrbU6YTYK/CfdfHbBcl1HP7R2RQgYCUTOG/rksc35LtLgXfAgED
o1EwTzALBgNVHQ8EBAMCAYYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQUJ/PI
FR5umgIJFq0roIlgX9p7L6owEAYJKwYBBAGCNxUBBAMCAQAwDQYJKoZIhvcNAQEF
BQADggEBAJ2dhISjQal8dwy3U8pORFBi71R803UXHOjgxkhLtv5MOhmBVrBW7hmW
Yqpao2TB9k5UM8Z3/sUcuuVdJcr18JOagxEu5sv4dEX+5wW4q+ffy0vhN4TauYuX
cB7w4ovXsNgOnbFp1iqRe6lJT37mjpXYgyc81WhJDtSd9i7rp77rMKSsH0T8lasz
Bvt9YAretIpjsJyp8qS5UwGH0GikJ3+r/+n6yUA4iGe0OcaEb1fJU9u6ju7AQ7L4
CYNu/2bPPu8Xs1gYJQk0XuPL1hS27PKSb3TkL4Eq1ZKR4OCXPDJoBYVL0fdX4lId
kxpUnwVwwEpxYB5DC2Ae/qPOgRnhCzU=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIEPDCCAySgAwIBAgIKYQlufQAAAAAADDANBgkqhkiG9w0BAQUFADA1MRYwFAYD
VQQKEw1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwNDgw
HhcNMTEwNjMwMTc1NjU3WhcNMjkwNTE0MjAyNTQyWjAnMQ4wDAYDVQQKEwVDaXNj
bzEVMBMGA1UEAxMMQUNUMiBTVURJIENBMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A
MIIBCgKCAQEA0m5l3THIxA9tN/hS5qR/6UZRpdd+9aE2JbFkNjht6gfHKd477AkS
5XAtUs5oxDYVt/zEbslZq3+LR6qrqKKQVu6JYvH05UYLBqCj38s76NLk53905Wzp
9pRcmRCPuX+a6tHF/qRuOiJ44mdeDYZo3qPCpxzprWJDPclM4iYKHumMQmqmgmg+
xghHIooWS80BOcdiynEbeP5rZ7qRuewKMpl1TiI3WdBNjZjnpfjg66F+P4SaDkGb
BXdGj13oVeF+EyFWLrFjj97fL2+8oauV43Qrvnf3d/GfqXj7ew+z/sXlXtEOjSXJ
URsyMEj53Rdd9tJwHky8neapszS+r+kdVQIDAQABo4IBWjCCAVYwCwYDVR0PBAQD
```

```
AgHGMB0GA1UdDgQWBBRI2PHxwnDVW7t8cwmTr7i4MAP4fzAfBgNVHSMEGDAWgBQn
88gVHm6aAgkWrSugiWBf2nsvqjBDBgNVHR8EPDA6MDigNqA0hjJodHRwOi8vd3d3
LmNpc2NvLmNvbS9zZWN1cml0eS9wa2kvY3JsL2NyY2EyMDQ4LmNybDBQBggrBgEF
BQcBAQREMEIwQAYIKwYBBQUHMAKGNGh0dHA6Ly93d3cuY2lzY28uY29tL3NlY3Vy
aXR5L3BraS9jZXJ0cy9jcmNhMjA0OC5jZXIwXAYDVR0gBFUwUzBRBgorBgEEAQkV
AQwAMEMwQQYIKwYBBQUHAgEWNWh0dHA6Ly93d3cuY2lzY28uY29tL3NlY3VyaXR5
L3BraS9wb2xpY2llcy9pbmRleC5odG1sMBIGA1UdEwEB/wQIMAYBAf8CAQAwDQYJ
KoZIhvcNAQEFBQADggEBAGh1qclr9tx4hzWgDERm371yeuEmqcIfi9b9+GbMSJbi
ZHc/CcCl0lJu0a9zTXA9w47H9/t6leduGxb4WeLxcwCiUgvFtCa51Iklt8nNbcKY
/4dw1ex+7amATUQO4QggIE67wVIPu6bgAE3Ja/nRS3xKYSnj8H5TehimBSv6TECi
i5jUhOWryAK4dVo8hCjkjEkzu3ufBTJapnv89g9OE+H3VKM4L+/KdkUO+52djFKn
hyl47d7cZR4DY4LIuFM2P1As8YyjzoNpK/urSRI14WdIlplR1nH7KNDl5618yfVP
0IFJZBGrooCRBjOSwFv8cpWCbmWdPaCQT2nwIjTfY8c=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIDgTCCAmmgAwIBAgIEAp4UYzANBgkqhkiG9w0BAQsFADAnMQ4wDAYDVQQKEwVD
aXNjbzEVMBMGA1UEAxMMQUNUMiBTVURJIENBMB4XDTE4MDYwNTAzNDUwNVoXDTI5
MDUxNDIwMjU0MVowbTEpMCcGA1UEBRMgUElEOkM5MjAwTC0yNFQtNEcgU046SlBH
MjIwMjAwQTgxDjAMBgNVBAoTBUNpc2NvMRgwFgYDVQQLEw9BQ1QtMiBMaXRlIFNV
REkxFjAUBgNVBAMTDUM5MjAwTC0yNFQtNEcwggEiMA0GCSqGSIb3DQEBAQUAA4IB
DwAwggEKAoIBAQDBm2Dg0GWQ18wLTKxeCt87DL8KlRbx8Db1IigHjzebBXMpx7Ja
6Cp+kwRrIWGi5AmNmV7jZ2ZLj+vFVzBQ9eGM+6LdNg18c6nqmSmnuXMerD1UEMMK
bkFl4ydn1EIMoWpCArbgz+/zaLM2A5bpQXVndiKq1v0NA2Pgvqdxbm+8AELdDG/D
3SQ1anOja+yH5vu3NjyMJfqtjzk+n/ILp9iZMWzcA+O6E8KC5FclR2cfvWlQvoFM
ZEWmHdhHPtsnN+4hhmDeurgeM0S+xIvzZq0H7PxS0kT4vYQ9xWQEwavJAL44k0uY
JxKP6bDNssSLZ2s4/2OBsODjyBhb0GwrOAHdAgMBAAGjbzBtMA4GA1UdDwEB/wQE
AwIF4DAMBgNVHRMBAf8EAjAAME0GA1UdEQRGMESgQgYJKwYBBAEJFQIDoDUTM0No
aXBJRD1RRGx6T0FZUHQwRTJJRVFFQUFjQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
PTANBgkqhkiG9w0BAQsFAAOCAQEAgLUxZfNmrXZ6ZMGX69dDPkmvp9cFqXR538LF
PdypCRuSk20GF8OeDUOsuIi4mbB87JSOWvLomdBtXdnxzRu4kPZNFz/7pjAVRT3R
gwMMyiEnDWQSvy7e4SZmyVgej55e3hTW/LTeU8lCE0KRoYGDce5Phv2zdHtIsXrV
XsY+Fropfntt1FV9qqDskDWcKf0bos6VsyWUpSCEGqF7LfNnBTKYvXUUmkXHKf/d
W5HgrYt6bQ/h/+0EP+MY2wpAiWMCfX6F+xW20vZfK8NzNesieB38IvuTkgefhz2s
yGCOavAxqGd0j7atcRpdrJt9+KM9Vwuy4VJZgK/t1fmTL4cawQ==
-----END CERTIFICATE-----

Signature version: 1
Signature:
[redacted]
```

The optional RSA 2048 signature is across the three certificates, the signature version and the user-provided nonce.

```
RSA PKCS#1v1.5 Sign {<Nonce (UINT64)> || <Signature Version (UINT32)> || <Cisco Root CA
2048 cert (DER)> ||
<Cisco subordinate CA (DER)> || <SUDI certificate (DER)> }
```

Cisco management solutions are equipped with the ability to interpret the above output. However, a simple script using OpenSSL commands can also be used to display the identity of the platform and to verify the signature, thereby ensuring its Cisco unique device identity.

```
[linux-host:~]openssl x509 -in sudicert.pem -subject -noout
subject= /serialNumber=PID:IE35xx-24T-4G SN:FDO1946BG05/O=Cisco/OU=ACT-2 Lite
SUDI/CN=IE35xx-24T-4G
```

Verifying Software Integrity

The following example displays the checksum record for the boot stages. The hash measurements are displayed for each of the three stages of software successively booted. These hashes can be compared against Cisco-provided reference values. An option to sign the output gives a verifier the ability to ensure the output is genuine and is not altered. A nonce can be provided to protect against replay attacks.

Note Boot integrity hashes are not MD5 hashes. For example, if you run **verify /md5 ie35xx_iosxe.17.17.01.SPA.bin** command for the bundle file, the hash will not match.

The following is a sample output of the **show platform integrity sign nonce 123** command in install mode. This output includes measurements of each installed package file.

```
Device#show platform integrity sign nonce 123
Platform: IE35xx-24T-4G
Boot 0 Version: SBOOT0.v27
Boot 0 Hash:
EE98DCD0D6AEA85C8891039F649664FCC3CF709CCFC7A6F248C9D5BA8463528F
Boot Loader Version: System Bootstrap, Version 10.2, DEVELOPMENT SOFTWARE
Boot Loader Hash:
922DB87E7A153A79EE9AE37311A1FDE2313C999E21032F8A1E7EF4935D3E7427657E4CDEE537E7B3C50E84121C00ED2556786A4EE155D3C0AFF67F63F1A69B
OS Version: 17.17.01
OS Hashes:
ie35xx_lite-rpbase.17.17.01.SPA.pkg :
DDD155C1DEFDB03EB0C64057AD6A973E2114EA7CCCAAA7ED0AE935CB0EB84E0DDD155C1DEFB03EB0C64057AD6A973E2114EA7CCCAAA7ED0AE935CB0EB84E0
ie35xx_lite-rpboot.17.17.01.SPA.pkg :
AD6A973E2114EA7CCCAAA7ED0AE935CB0EB84E0DDD155C1DEFDB03EB0C64057AD6A973E2114EA7CCCAAA7ED0AE935CB0EB84E0DDD155C1DEFB03EB0C64057
ie35xx_lite-srdriver.17.17.01.SPA.pkg :
4EA7CCCAAA7ED0AE935CB0EB84E0DDD155C1DEFDB03EB0C64057AD6A973E2114EA7CCCAAA7ED0AE935CB0EB84E0DDD155C1DEFB03EB0C64057AD6A973E211
ie35xx_lite-webui.17.17.01.SPA.pkg :
CCCAAA7ED0AE935CB0EB84E0DDD155C1DEFDB03EB0C64057AD6A973E2114EA7CCCAAA7ED0AE935CB0EB84E0DDD155C1DEFB03EB0C64057AD6A973E2114EA7
ie35xx-wlc.17.17.01.SPA.pkg :
AA7ED0AE935CB0EB84E0DDD155C1DEFDB03EB0C64057AD6A973E2114EA7CCCAAA7ED0AE935CB0EB84E0DDD155C1DEFB03EB0C64057AD6A973E2114EA7CCCA
PCR0: 750E5D2EDAE6E3A68050638E0BFD8619BE4EA13066025D39DF79408719F5177E
PCR8: EB6E739A63F53E703B6CDAF3F6188833CEF6D32E2F726006B9AA34E1E73048C4
Signature version: 1
Signature:
[redacted]
```

The following is a sample output of the **show platform integrity sign nonce 123** command in bundle mode. This output includes measurements of the bundle file and each installed package.

```
Device# show platform integrity sign nonce 123
Platform: IE35xx-24T-4G
Boot 0 Version: SBOOT0.v27
Boot 0 Hash:
EE98DCD0D6AEA85C8891039F649664FCC3CF709CCFC7A6F248C9D5BA8463528F
Boot Loader Version: System Bootstrap, Version 10.2, DEVELOPMENT SOFTWARE
Boot Loader Hash:
922DB87E7A153A79EE9AE37311A1FDE2313C999E21032F8A1E7EF4935D3E7427657E4CDEE537E7B3C50E84121C00ED2556786A4EE155D3C0AFF67F63F1A69B
OS Version: 17.17.01
OS Hashes:
ie35xx_lite_iosxe.17.17.01.SPA.bin :
F4CAD08BE1EF841C3A2E3ED854082F08F3CBA9336F38E45669D4D8B15AD15E36EB922AC8B4D0D5B63E2806D6A1BDAB7839D9D08CD7E366A49ED648C113440
ie35xx_lite-rpbase.17.17.01.SPA.pkg :
DDD155C1DEFDB03EB0C64057AD6A973E2114EA7CCCAAA7ED0AE935CB0EB84E0DDD155C1DEFB03EB0C64057AD6A973E2114EA7CCCAAA7ED0AE935CB0EB84E0
ie35xx_lite-rpboot.17.17.01.SPA.pkg :
```

```
AD6A9673E2114EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057
ie35xx_lite-srdriver.17.17.01.SPA.pkg :
4EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E211
ie35xx_lite-webui.17.17.01.SPA.pkg :
CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7
ie35xx-wlc.17.17.01.SPA.pkg :
AA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0EB84E0DD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCA
PCR0: 750E5D2EDAE6E3A68050638E0BFD8619BE4EA13066025D39DF79408719F5177E
PCR8: EB6E739A63F53E703B6CDAF3F6188833CEF6D32E2F726006B9AA34E1E73048C4
Signature version: 1
Signature:
[redacted]
```

Verifying Image Signing

The following example displays the secure code signing check of the image during bootup using an SHA-512 hash.

```
switch:boot flash:packages.conf
boot: attempting to boot from [flash:packages.conf]
boot: reading file packages.conf
#
Performing Integrity Check ...
boot: parsed image from conf file: ie35xx-rpboot.17.17.1.SSA.pkg
[redacted]


Loading image in Verbose mode: 1


Image Base is: 0x100099000
Image Size is: 0x2C83487
Package header rev 3 structure detected
Package type:30001, flags:0x0
IsoSize = 0
Parsing package TLV info:
000: 000000090000001D4B45595F544C565F -         KEY_TLV_
010: 5041434B4147455F434F4D5041544942 - PACKAGE_COMPATIB
020: 494C4954590000000000000090000000B - ILITY
030: 4652555F52505F545950450000000009 - FRU_RP_TYPE
040: 000000184B45595F544C565F5041434B -     KEY_TLV_PACK
050: 4147455F424F4F544152434800000009 - AGE_BOOTARCH
060: 0000000E415243485F693638365F5459 -     ARCH_i686_TY
070: 5045000000000009000000144B45595F - PE          KEY_
080: 544C565F424F4152445F434F4D504154 - TLV_BOARD_COMPAT
090: 0000000900000010424F4152445F6361 -         BOARD_ca
0A0: 74396B5F5459504500000000900000018 - t9k_TYPE
0B0: 4B45595F544C565F43525950544F5F4B - KEY_TLV_CRYPTO_K
0C0: 4559535452494E470000000900000004 - EYSTRING

TLV: T=9, L=29, V=KEY_TLV_PACKAGE_COMPATIBILITY
TLV: T=9, L=11, V=FRU_RP_TYPE
TLV: T=9, L=24, V=KEY_TLV_PACKAGE_BOOTARCH
TLV: T=9, L=14, V=ARCH_i686_TYPE
TLV: T=9, L=20, V=KEY_TLV_BOARD_COMPAT
```

```
TLV: T=9, L=16, V=BOARD_ie35xx_TYPE
TLV: T=9, L=24, V=KEY_TLV_CRYPTO_KEYSTRING
TLV: T=9, L=4, V=none
TLV: T=9, L=11, V=CW_BEGIN=$$
TLV: T=9, L=17, V=CW_FAMILY=$ie35xx$
TLV: T=9, L=74, V=CW_IMAGE=$ie35xx-rpboot.17.17.1.SSA.pkg$
TLV: T=9, L=20, V=CW_VERSION=$X.X.X$
IOS version is X.X.X
TLV: T=9, L=53, V=CW_FULL_VERSION=$17.17.1$
TLV: T=9, L=52, V=CW_DESCRIPTION=$Cisco IOS Software, IOS-XE Software$
TLV: T=9, L=9, V=CW_END=$$
Found DIGISIGN TLV type 12 length = 392
RSA Self Test Passed

Expected hash:
DDAF35A193617ABACC417349AE204131
12E6FA4E89A97EA20A9EEEE64B55D39A
2192992A274FC1A836BA3C23A3FEEBBD
454D4423643CE80E2A9AC94FA54CA49F

Obtained hash:
DDAF35A193617ABACC417349AE204131
12E6FA4E89A97EA20A9EEEE64B55D39A
2192992A274FC1A836BA3C23A3FEEBBD
454D4423643CE80E2A9AC94FA54CA49F
Sha512 Self Test Passed
Found package arch type ARCH_i686_TYPE
Found package FRU type FRU_RP_TYPE
Performing Integrity Check ...

RSA Signed DEVELOPMENT Image Signature Verification Successful.
```

CHAPTER 136

Performing Device Setup Configuration

Restrictions for Performing Device Setup Configuration

- Subpackage software installation is not supported.

Information About Performing Device Setup Configuration

The following sections provide information about how to perform a device setup configuration, including IP address assignments and Dynamic Host Configuration Protocol (DHCP) auto configuration.

Device Boot Process

To start your device, you need to follow the procedures described in the Hardware Installation Guide for installing and powering on the device and setting up the initial device configuration.

The normal boot process involves the operation of the boot loader software and includes these activities:

- Performs low-level CPU initialization. This process initializes the CPU registers that control where physical memory is mapped, the quantity and speed of the physical memory, and so forth.
- Initializes the file systems on the system board.
- Loads a default operating system software image into memory and boots up the device.
- Performs power-on self-test (POST) for the CPU subsystem and tests the system DRAM. As part of POST, the following tests are also performed:
  - MAC loopback test to verify the data path between the CPU and network ports.
  - Power over Ethernet (PoE) controller functionality test to check the chip accessibility, firmware download, and health status of the power-sourcing equipment.
  - Thermal test to verify the temperature reading from the device sensor.

For information about the complete list of supported online diagnostics, see the Configuring Online Diagnostics chapter.

The boot loader provides access to the file systems before the operating system is loaded. Normally, the boot loader is used only to load, decompress, and start the operating system. After the boot loader gives the operating system control of the CPU, the boot loader is not active until the next system reset or power-on.

Before you can assign device information, make sure you have connected a PC or terminal to the console port or a PC to the Ethernet management port, and make sure you have configured the PC or terminal-emulation software baud rate and character format to match these of the device console port:

- Baud rate default is 9600.
- Data bits default is 8.

Note If the data bits option is set to 8, set the parity option to none.

- Stop bits default is 2 (minor).
- Parity settings default is none.

Software Install Overview

The Software Install feature provides a uniform experience across different types of upgrades, such as full image install, Software Maintenance Upgrade (SMU), In-Service Software Upgrade (ISSU)and In-Service Model Update (data model package).

The Software Install feature facilitates moving from one version of the software to another version in install mode. Use the **install** command in privileged EXEC mode to install or upgrade a software image. You can also downgrade to a previous version of the software image, using the install mode.

The method that you use to upgrade Cisco IOS XE software depends on whether the switch is running in install mode or in bundle mode. In bundle mode or consolidated boot mode, a .bin image file is used from a local or remote location to boot the device. In the install boot mode, the bootloader uses the packages.conf file to boot up the device.

The following software install features are supported on your switch:

- Software bundle installation on a standalone switch.
- Software rollback to a previously installed package set.

Software Boot Modes

Your device supports two modes to boot the software packages:

Installed Boot Mode

You can boot your switch in installed mode by booting the software package provisioning file that resides in flash:

```
Switch: boot flash:packages.conf
```

Note We recommend that you use the install mode for Cisco IE3500 Series Switches.

Note The packages.conf file for particular release is created on following the install workflow described in the section, *Installing a Software Package*.

The provisioning file contains a list of software packages to boot, mount, and run. The ISO file system in each installed package is mounted to the root file system directly from flash.

Note The packages and provisioning file used to boot in installed mode must reside in flash. Booting in installed mode from usbflash0: or tftp: is not supported.

Bundle Boot Mode

You can boot your device in bundle boot mode by booting the bundle (.bin) file:

```
switch:  ie35xx-17.17.1.spa.bin
```

The provisioning file contained in a bundle is used to decide which packages to boot, mount, and run. Packages are extracted from the bundle and copied to RAM. The ISO file system in each package is mounted to the root file system.

Unlike install boot mode, additional memory that is equivalent to the size of the bundle is used when booting in bundle mode.

Unlike install boot mode, bundle boot mode is available from several locations:

- flash:
- usbflash0:
- tftp:

Changing the Boot Mode

To change a device running in bundle boot mode to install mode, set the boot variable to flash:packages.conf, and execute the **install add file flash:ie35xx-17.17.1.spa.bin activate commit** command. After the command is executed, the device reboots in install boot mode.

Installing the Software Package

You can install the software package on a device by using the **install add** commands in privileged EXEC mode.

The **install add** command copies the software package from a local or remote location to the device. The location can be FTP, HTTP, HTTPs, or TFTP. The command extracts individual components of the .bin file into sub-packages and packages.conf file. It also validates the file to ensure that the image file is specific to the platform.

Note The install operation through SCP is not supported.

Terminating a Software Install

You can terminate the activation of a software image in the following ways:

- Using the **install activate auto-abort-timer** command. When the device reloads after activating a new image, the auto-abort-timer is triggered. If the timer expires before issuing the **install commit** command, then the installation process is terminated; the device reloads again and boots up with the previous version of the software image.

  Use the **install auto-abort-timer stop** command to stop this timer.

- Using the **install abort** command. This command rolls back to the version that was running before installing the new software. Use this command before issuing the **install commit** command.

Devices Information Assignment

You can assign IP information through the device setup program, through a DHCP server, or manually.

Use the device setup program if you want to be prompted for specific IP information. With this program, you can also configure a hostname and an enable secret password. For a new switch, enter a new password for enable secret password.

It gives you the option of assigning a Telnet password (to provide security during remote management) and configuring your switch as a command or member switch of a cluster or as a standalone switch.

Use a DHCP server for centralized control and automatic assignment of IP information after the server is configured.

Note If you are using DHCP, do not respond to any of the questions in the setup program until the device receives the dynamically assigned IP address and reads the configuration file.

If you are an experienced user familiar with the device configuration steps, manually configure the device. Otherwise, use the setup program described in section Device Boot Process, on page 2015.

Default Switch Information

Table 142: Default Switch Information

Feature	Default Setting
IP address and subnet mask	No IP address or subnet mask are defined.
Default gateway	No default gateway is defined.
Enable secret password	No password is defined.

Feature	Default Setting
Hostname	The factory-assigned default hostname is device.
Telnet password	No password is defined.
Cluster command switch functionality	Disabled.
Cluster name	No cluster name is defined.

DHCP-Based Autoconfiguration Overview

DHCP provides configuration information to Internet hosts and internetworking devices. This protocol consists of two components: one for delivering configuration parameters from a DHCP server to a device and an operation for allocating network addresses to devices. DHCP is built on a client-server model, in which designated DHCP servers allocate network addresses and deliver configuration parameters to dynamically configured devices. The device can act as both a DHCP client and a DHCP server.

During DHCP-based autoconfiguration, your device (DHCP client) is automatically configured at startup with IP address information and a configuration file.

With DHCP-based autoconfiguration, no DHCP client-side configuration is needed on your device. However, you need to configure the DHCP server for various lease options associated with IP addresses.

If you want to use DHCP to relay the configuration file location on the network, you might also need to configure a Trivial File Transfer Protocol (TFTP) server and a Domain Name System (DNS) server.

The DHCP server for your device can be on the same LAN or on a different LAN than the device. If the DHCP server is running on a different LAN, you should configure a DHCP relay device between your device and the DHCP server. A relay device forwards broadcast traffic between two directly connected LANs. A router does not forward broadcast packets, but it forwards packets based on the destination IP address in the received packet.

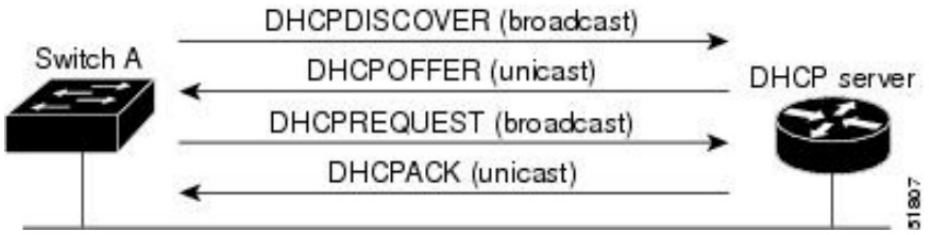
DHCP-based autoconfiguration replaces the BOOTP client functionality on your device.

DHCP Client Request Process

When you boot up your device, the DHCP client is invoked and requests configuration information from a DHCP server when the configuration file is not present on the device. If the configuration file is present and the configuration includes the **ip address dhcp** interface configuration command on specific routed interfaces, the DHCP client is invoked and requests the IP address information for those interfaces.

This is the sequence of messages that are exchanged between the DHCP client and the DHCP server.

Figure 142: DHCP Client and Server Message Exchange



The client, Device A, broadcasts a DHCPDISCOVER message to locate a DHCP server. The DHCP server offers configuration parameters (such as an IP address, subnet mask, gateway IP address, DNS IP address, a lease for the IP address, and so forth) to the client in a DHCPOFFER unicast message.

In a DHCPREQUEST broadcast message, the client returns a formal request for the offered configuration information to the DHCP server. The formal request is broadcast so that all other DHCP servers that received the DHCPDISCOVER broadcast message from the client can reclaim the IP addresses that they offered to the client.

The DHCP server confirms that the IP address has been allocated to the client by returning a DHCPACK unicast message to the client. With this message, the client and server are bound, and the client uses configuration information received from the server. The amount of information the device receives depends on how you configure the DHCP server.

If the configuration parameters sent to the client in the DHCPOFFER unicast message are invalid (a configuration error exists), the client returns a DHCPDECLINE broadcast message to the DHCP server.

The DHCP server sends the client a DHCPNAK denial broadcast message, which means that the offered configuration parameters have not been assigned, that an error has occurred during the negotiation of the parameters, or that the client has been slow in responding to the DHCPOFFER message (the DHCP server assigned the parameters to another client).

A DHCP client might receive offers from multiple DHCP or BOOTP servers and can accept any of the offers; however, the client usually accepts the first offer it receives. The offer from the DHCP server is not a guarantee that the IP address is allocated to the client; however, the server usually reserves the address until the client has had a chance to formally request the address. If the device accepts replies from a BOOTP server and configures itself, the device broadcasts, instead of unicasts, TFTP requests to obtain the device configuration file.

The DHCP hostname option allows a group of devices to obtain hostnames and a standard configuration from the central management DHCP server. A client (device) includes in its DCHPDISCOVER message an option 12 field used to request a hostname and other configuration parameters from the DHCP server. The configuration files on all clients are identical except for their DHCP-obtained hostnames.

DHCP-Based Autoconfiguration and Image Update

You can use the DHCP image upgrade features to configure a DHCP server to download both a new image and a new configuration file to one or more devices in a network. Simultaneous image and configuration upgrade for all switches in the network helps ensure that each new device added to a network receives the same image and configuration.

There are two types of DHCP image upgrades: DHCP autoconfiguration and DHCP auto-image update.

Restrictions for DHCP-Based Autoconfiguration

- The DHCP-based autoconfiguration with a saved configuration process stops if there is not at least one Layer 3 interface in an up state without an assigned IP address in the network.
- Unless you configure a timeout, the DHCP-based autoconfiguration with a saved configuration feature tries indefinitely to download an IP address.
- The auto-install process stops if a configuration file cannot be downloaded or if the configuration file is corrupted.
- The configuration file that is downloaded from TFTP is merged with the existing configuration in the running configuration but is not saved in the NVRAM unless you enter the **write memory** or **copy running-configuration startup-configuration** privileged EXEC command. If the downloaded configuration is saved to the startup configuration, the feature is not triggered during subsequent system restarts.

DHCP Autoconfiguration

DHCP autoconfiguration downloads a configuration file to one or more devices in your network from a DHCP server. The downloaded configuration file becomes the running configuration of the device. It does not over write the bootup configuration saved in the flash, until you reload the device.

DHCP Auto-Image Update

You can use DHCP auto-image upgrade with DHCP autoconfiguration to download both a configuration and a new image to one or more devices in your network. The devices (or devices) downloading the new configuration and the new image can be blank (or only have a default factory configuration loaded).

If the new configuration is downloaded to a switch that already has a configuration, the downloaded configuration is appended to the configuration file stored on the switch. (Any existing configuration is not overwritten by the downloaded one.)

To enable a DHCP auto-image update on the device, the TFTP server where the image and configuration files are located must be configured with the correct option 67 (the configuration filename), option 66 (the DHCP server hostname) option 150 (the TFTP server address), and option 125 (description of the Cisco IOS image file) settings.

After you install the device in your network, the auto-image update feature starts. The downloaded configuration file is saved in the running configuration of the device, and the new image is downloaded and installed on the device. When you reboot the device, the configuration is stored in the saved configuration on the device.

DHCP Server Configuration Guidelines

Follow these guidelines if you are configuring a device as a DHCP server:

- You should configure the DHCP server with reserved leases that are bound to each device by the device hardware address.
- If you want the device to receive IP address information, you must configure the DHCP server with these lease options:
  - IP address of the client (required)
  - Subnet mask of the client (required)

- DNS server IP address (optional)
- Router IP address (default gateway address to be used by the device) (required)

- If you want the device to receive the configuration file from a TFTP server, you must configure the DHCP server with these lease options:
  - TFTP server name (required)
  - Boot filename (the name of the configuration file that the client needs) (recommended)
  - Hostname (optional)

- Depending on the settings of the DHCP server, the device can receive IP address information, the configuration file, or both.
- If you do not configure the DHCP server with the lease options described previously, it replies to client requests with only those parameters that are configured. If the IP address and the subnet mask are not in the reply, the device is not configured. If the router IP address or the TFTP server name are not found, the device might send broadcast, instead of unicast, TFTP requests. Unavailability of other lease options does not affect autoconfiguration.
- The device can act as a DHCP server. By default, the Cisco IOS DHCP server and relay agent features are enabled on your device but are not configured. (These features are not operational.)

Purpose of the TFTP Server

Based on the DHCP server configuration, the device attempts to download one or more configuration files from the TFTP server. If you configured the DHCP server to respond to the device with all the options required for IP connectivity to the TFTP server, and if you configured the DHCP server with a TFTP server name, address, and configuration filename, the device attempts to download the specified configuration file from the specified TFTP server.

If you did not specify the configuration filename, the TFTP server, or if the configuration file could not be downloaded, the device attempts to download a configuration file by using various combinations of filenames and TFTP server addresses. The files include the specified configuration filename (if any) and these files: network-config, cisconet.cfg, *hostname*.config, or *hostname*.cfg, where *hostname* is the device's current hostname. The TFTP server addresses used include the specified TFTP server address (if any) and the broadcast address (255.255.255.255).

For the device to successfully download a configuration file, the TFTP server must contain one or more configuration files in its base directory. The files can include these files:

- The configuration file named in the DHCP reply (the actual device configuration file).
- The network-confg or the cisconet.cfg file (known as the default configuration files).
- The router-confg or the ciscortr.cfg file (These files contain commands common to all device. Normally, if the DHCP and TFTP servers are properly configured, these files are not accessed.)

If you specify the TFTP server name in the DHCP server-lease database, you must also configure the TFTP server name-to-IP-address mapping in the DNS-server database.

If the TFTP server to be used is on a different LAN from the device, or if it is to be accessed by the device through the broadcast address (which occurs if the DHCP server response does not contain all the required

information described previously), a relay must be configured to forward the TFTP packets to the TFTP server. The preferred solution is to configure the DHCP server with all the required information.

Purpose of the DNS Server

The DHCP server uses the DNS server to resolve the TFTP server name to an IP address. You must configure the TFTP server name-to-IP address map on the DNS server. The TFTP server contains the configuration files for the device.

You can configure the IP addresses of the DNS servers in the lease database of the DHCP server from where the DHCP replies will retrieve them. You can enter up to two DNS server IP addresses in the lease database.

The DNS server can be on the same LAN or on a different LAN from the device. If it is on a different LAN, the device must be able to access it through a router.

How to Obtain Configuration Files

Depending on the availability of the IP address and the configuration filename in the DHCP reserved lease, the device obtains its configuration information in these ways:

- The IP address and the configuration filename is reserved for the device and provided in the DHCP reply (one-file read method).

  The device receives its IP address, subnet mask, TFTP server address, and the configuration filename from the DHCP server. The device sends a unicast message to the TFTP server to retrieve the named configuration file from the base directory of the server and upon receipt, it completes its boot up process.

- The IP address and the configuration filename is reserved for the device, but the TFTP server address is not provided in the DHCP reply (one-file read method).

  The device receives its IP address, subnet mask, and the configuration filename from the DHCP server. The device sends a broadcast message to a TFTP server to retrieve the named configuration file from the base directory of the server, and upon receipt, it completes its boot-up process.

- Only the IP address is reserved for the device and provided in the DHCP reply. The configuration filename is not provided (two-file read method).

  The device receives its IP address, subnet mask, and the TFTP server address from the DHCP server. The device sends a unicast message to the TFTP server to retrieve the network-confg or cisconet.cfg default configuration file. (If the network-confg file cannot be read, the device reads the cisconet.cfg file.)

  The default configuration file contains the hostnames-to-IP-address mapping for the device. The device fills its host table with the information in the file and obtains its hostname. If the hostname is not found in the file, the device uses the hostname in the DHCP reply. If the hostname is not specified in the DHCP reply, the device uses the default *Switch* as its hostname.

  After obtaining its hostname from the default configuration file or the DHCP reply, the device reads the configuration file that has the same name as its hostname (*hostname*-confg or *hostname*.cfg, depending on whether network-confg or cisconet.cfg was read earlier) from the TFTP server. If the cisconet.cfg file is read, the filename of the host is truncated to eight characters.

  If the device cannot read the network-confg, cisconet.cfg, or the hostname file, it reads the router-confg file. If the device cannot read the router-confg file, it reads the ciscortr.cfg file.

Note The device broadcasts TFTP server requests if the TFTP server is not obtained from the DHCP replies, if all attempts to read the configuration file through unicast transmissions fail, or if the TFTP server name cannot be resolved to an IP address.

How to Control Environment Variables

With a normally operating device, you enter the boot loader mode only through the console connection configured for 9600 bps. Unplug the device power cord, and press the **Mode** button while reconnecting the power cord. The boot loader device prompt then appears.

The device boot loader software provides support for nonvolatile environment variables, which can be used to control how the boot loader, or any other software running on the system, operates. Boot loader environment variables are similar to environment variables that can be set on UNIX or DOS systems.

Environment variables that have values are stored in flash memory outside of the flash file system.

Each line in these files contains an environment variable name and an equal sign followed by the value of the variable. A variable has no value if it is not present; it has a value if it is listed even if the value is a null string. A variable that is set to a null string (for example, " ") is a variable with a value. Many environment variables are predefined and have default values.

You can change the settings of the environment variables by accessing the boot loader or by using Cisco IOS commands. Under normal circumstances, it is not necessary to alter the setting of the environment variables.

Scheduled Reload of the Software Image

You can schedule a reload of the software image to occur on the device at a later time (for example, late at night or during the weekend when the device is used less), or you can synchronize a reload network-wide (for example, to perform a software upgrade on all device in the network).

Note A scheduled reload must take place within approximately 24 days.

You have these reload options:

- Reload of the software to take affect in the specified minutes or hours and minutes. The reload must take place within approximately 24 hours. You can specify the reason for the reload in a string up to 255 characters in length.
- Reload of the software to take place at the specified time (using a 24-hour clock). If you specify the month and day, the reload is scheduled to take place at the specified time and date. If you do not specify the month and day, the reload takes place at the specified time on the current day (if the specified time is later than the current time) or on the next day (if the specified time is earlier than the current time). Specifying 00:00 schedules the reload for midnight.

The **reload** command halts the system. If the system is not set to manually boot up, it reboots itself.

If your device is configured for manual booting, do not reload it from a virtual terminal. This restriction prevents the device from entering the boot loader mode and then taking it from the remote user's control.

If you modify your configuration file, the device prompts you to save the configuration before reloading. During the save operation, the system requests whether you want to proceed with the save if the CONFIG_FILE environment variable points to a startup configuration file that no longer exists. If you proceed in this situation, the system enters setup mode upon reload.

To cancel a previously scheduled reload, use the **reload cancel** privileged EXEC command.

How to Perform Device Setup Configuration

Using DHCP to download a new image and a new configuration to a device requires that you configure at least two devices. One device acts as a DHCP and TFTP server and the second device (client) is configured to download either a new configuration file or a new configuration file and a new image file.

Configuring DHCP Autoconfiguration (Only Configuration File)

This task describes how to configure DHCP autoconfiguration of the TFTP and DHCP settings on an existing device in the network so that it can support the autoconfiguration of a new device.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**ip dhcp pool** *poolname* **Example:** `Device(config)# ip dhcp pool pool`	Creates a name for the DHCP server address pool, and enters DHCP pool configuration mode.
Step 3	**boot** *filename* **Example:** `Device(dhcp-config)# boot config-boot.text`	Specifies the name of the configuration file that is used as a boot image.
Step 4	**network** *network-number mask prefix-length* **Example:** `Device(dhcp-config)# network 10.10.10.0 255.255.255.0`	Specifies the subnet network number and mask of the DHCP address pool. **Note** The prefix length specifies the number of bits that comprise the address prefix. The prefix is an alternative way of specifying the network mask of the client. The prefix length must be preceded by a forward slash (/).

	Command or Action	Purpose
Step 5	**default-router** *address* **Example:** `Device(dhcp-config)# default-router 10.10.10.1`	Specifies the IP address of the default router for a DHCP client.
Step 6	**option 150** *address* **Example:** `Device(dhcp-config)# option 150 10.10.10.1`	Specifies the IP address of the TFTP server.
Step 7	**exit** **Example:** `Device(dhcp-config)# exit`	Returns to global configuration mode.
Step 8	**tftp-server flash**:*filename.text* **Example:** `Device(config)# tftp-server flash:config-boot.text`	Specifies the configuration file on the TFTP server.
Step 9	**interface** *interface-id* **Example:**	Specifies the address of the client that will receive the configuration file.
Step 10	**no switchport** **Example:** `Device(config-if)# no switchport`	Puts the interface into Layer 3 mode.
Step 11	**ip address** *address mask* **Example:** `Device(config-if)# ip address 10.10.10.1 255.255.255.0`	Specifies the IP address and mask for the interface.
Step 12	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Manually Assigning IP Information to Multiple SVIs

This task describes how to manually assign IP information to multiple switched virtual interfaces (SVIs):

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface vlan** *vlan-id* **Example:** `Device(config)# interface vlan 99`	Enters interface configuration mode, and enters the VLAN to which the IP information is assigned. The range is 1 to 4094.
Step 4	**ip address** *ip-address subnet-mask* **Example:** `Device(config-vlan)# ip address 10.10.10.2 255.255.255.0`	Enters the IP address and subnet mask.
Step 5	**exit** **Example:** `Device(config-vlan)# exit`	Returns to global configuration mode.
Step 6	**ip default-gateway** *ip-address* **Example:** `Device(config)# ip default-gateway 10.10.10.1`	Enters the IP address of the next-hop router interface that is directly connected to the device where a default gateway is being configured. The default gateway receives IP packets with unresolved destination IP addresses from the device. Once the default gateway is configured, the device has connectivity to the remote networks with which a host needs to communicate. **Note**

	Command or Action	Purpose
		When your device is configured to route with IP, it does not need to have a default gateway set. **Note** The device capwap relays on default-gateway configuration to support routed access point join the device.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show interfaces vlan** *vlan-id* **Example:** `Device# show interfaces vlan 99`	Displays the interfaces status for the specified VLAN.
Step 9	**show ip redirects** **Example:** `Device# show ip redirects`	Displays the Internet Control Message Protocol (ICMP) redirect messages.

Modifying Device Startup Configuration

The following sections provide information on how to modify the startup configuration of a device.

Specifying a Filename to Read and Write a System Configuration

By default, the Cisco IOS software uses the config.text file to read and write a nonvolatile copy of the system configuration. However, you can specify a different filename, which will be loaded during the next boot cycle.

Before you begin

Use a standalone device for this task.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**boot flash**:*/file-url* **Example:** `Device(config)# boot flash:config.text`	Specifies the configuration file to load during the next boot cycle. • *file-url*: The path (directory) and the configuration filename. • Filenames and directory names are case-sensitive.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show boot** **Example:** `Device# show boot`	Lists the contents of the BOOT environment variable (if set), the name of the configuration file pointed to by the CONFIG_FILE environment variable, and the contents of the BOOTLDR environment variable. • The **boot** global configuration command changes the setting of the CONFIG_FILE environment variable.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Booting the Device in Installed Mode

Installing a Software Package

You can install, activate, and commit a software package using a single command or using separate commands. This task shows how to use the **install add file activate commit** command for installing a software package.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**install add file tftp:** *filename* [**activate commit**] **Example:** `Device# install add file flash:ie35xx-17.17.1.spa.bin activate commit`	Copies the software install package from a remote location (via FTP, HTTP, HTTPs, TFTP) to the device, performs a compatibility check for the platform and image versions, activates the software package, and makes the package persistent across reloads. • This command extracts the individual components of the .bin file into sub-packages and packages.conf file. • The device reloads after executing this command.
Step 3	**exit** **Example:** `Device# exit`	Exits privileged EXEC mode and returns to user EXEC mode.

Managing the Update Package

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**install add file tftp:** *filename* **Example:** `Device# install add file tftp://172.16.0.1//tftpboot/folder1/ie35xx-17.17.1.spa.bin`	Copies the software install package from a remote location (via FTP, HTTP, HTTPs, TFTP) to the device, and performs a compatibility check for the platform and image versions. • This command extracts the individual components of the .bin file into sub-packages and packages.conf file.
Step 3	**install activate** [**auto-abort-timer**] **Example:**	Activates the added software install package, and reloads the device.

	Command or Action	Purpose
	`Device# install activate`	• When doing a full software install, do not provide a package filename. • The **auto-abort-timer** keyword, automatically rolls back the software image activation. The automatic timer is triggered after the new image is activated. If the timer expires prior to the issuing of the **install commit** command, then the install process is automatically terminated. The device reloads, and boots up with a previous version of the software image.
Step 4	**install abort** **Example:** `Device# install abort`	(Optional) Terminates the software install activation, and rolls back to the version that was running before current installation procedure. • You can use this command only when the image is in an activated state; and not when the image is in a committed state.
Step 5	**install commit** **Example:** `Device# install commit`	Makes the changes persistent over reload. • The **install commit** command completes the new image installation. Changes are persistent across reloads until the auto-abort timer expires.
Step 6	**install rollback to committed** **Example:** `Device# install rollback to committed`	(Optional) Rolls back the update to the last committed version.
Step 7	**install remove** {**file** *filesystem: filename* | **inactive**} **Example:** `Device# install remove inactive`	(Optional) Deletes all unused and inactive software installation files.
Step 8	**show install summary** **Example:** `Device# show install summary`	Displays information about the active package. • The output of this command varies according to the **install** commands that are configured.

Booting a Device in Bundle Mode

There are several methods by which you can boot the device — either by copying the bin file from the TFTP server and then boot the device, or by booting the device straight from flash or USB flash using the commands **boot flash:<image.bin>** or **boot usbflash0:<image.bin>** .

The following procedure explains how to boot the device from the TFTP server in the bundle mode.

Procedure

	Command or Action	Purpose
Step 1	**switch:BOOT=**<*source path of .bin file*> **Example:** `switch:BOOT=tftp://10.0.0.2/ie35xx-17.17.1.spa.bin`	Sets the boot parameters.
Step 2	**boot** **Example:** `switch:boot`	Boots the device.
Step 3	**show version**	(Optional) Displays the version of the image installed.

Configuring a Scheduled Software Image Reload

This task describes how to configure your device to reload the software image at a later time.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves your device configuration information to the startup configuration before you use the **reload** command.

	Command or Action	Purpose
Step 4	**reload in** [*hh*:]*mm* [*text*] **Example:** `Device# reload in 12` `System configuration has been modified. Save? [yes/no]: y`	Schedules a reload of the software to take affect in the specified minutes or hours and minutes. The reload must take place within approximately 24 days. You can specify the reason for the reload in a string up to 255 characters in length.
Step 5	**reload at** *hh*: *mm* [*month day* | *day month*] [*text*] **Example:** `Device(config)# reload at 14:00`	Specifies the time in hours and minutes for the reload to occur. **Note** Use the **at** keyword only if the device system clock has been set (through Network Time Protocol (NTP), the hardware calendar, or manually). The time is relative to the configured time zone on the device. To schedule reloads across several devices to occur simultaneously, the time on each device must be synchronized with NTP.
Step 6	**reload cancel** **Example:** `Device(config)# reload cancel`	Cancels a previously scheduled reload.
Step 7	**show reload** **Example:** `show reload`	Displays information about a previously scheduled reload or identifies if a reload has been scheduled on the device.

Configuration Examples for Device Setup Configuration

The following sections provide configuration examples for device setup.

Examples: Displaying Software Bootup in Install Mode

The following example displays software bootup in install mode:

```
switch: boot flash:packages.conf
Attempting to boot from [flash:packages.conf]
Located packages.conf
#

validate_package: SHA-1 hash:
        expected 340D5091:2872A0DD:03E9068C:3FDBECAB:69786462
        calculated 340D5091:2872A0DD:03E9068C:3FDBECAB:69786462
Image parsed from conf file is ie35xx-rpboot.17.17.01.SPA.pkg
##################################################################################################################
```

```
Waiting for 120 seconds for other switches to boot
##############################################################################################################
Switch number is 1

              Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706



Cisco IOS Software, L3 Switch Software (ie35xx_IOSXE), Version 17.17.1, RELEASE SOFTWARE
(fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2025 by Cisco Systems, Inc.
Compiled Tue 30-Oct-25 00:36 by mcpre



Cisco IOS-XE software, Copyright (c) 2005-2025 by cisco Systems, Inc.
All rights reserved.  Certain components of Cisco IOS-XE software are
licensed under the GNU General Public License ("GPL") Version 2.0.  The
software code licensed under GPL Version 2.0 is free software that comes
with ABSOLUTELY NO WARRANTY.  You can redistribute and/or modify such
GPL code under the terms of GPL Version 2.0.  For more details, see the
documentation or "License Notice" file accompanying the IOS-XE software,
or the applicable URL provided on the flyer accompanying the IOS-XE
software.



FIPS: Flash Key Check : Begin
FIPS: Flash Key Check : End, Not Found, FIPS Mode Not Enabled

This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html

If you require further assistance please contact us by sending email to
export@cisco.com.
cisco IE35xx-25P-4G (ARM64) processor with 518473K/3071K bytes of memory.
Processor board ID JPG221000RH
988 Virtual Ethernet interfaces
56 Gigabit Ethernet interfaces
2048K bytes of non-volatile configuration memory.
```

```
2015456K bytes of physical memory.
819200K bytes of Crash Files at crashinfo:.
1941504K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.
819200K bytes of Crash Files at crashinfo-7:.
1941504K bytes of Flash at flash-7:.

Base Ethernet MAC Address          : 68:2c:7b:f7:49:00
Motherboard Assembly Number        : 73-18699-2
Motherboard Serial Number          : JAE22090AZB
Model Revision Number              : 13
Motherboard Revision Number        : 05
Model Number                       : IE35xx-25P-4G
System Serial Number               : JPG221000RH


%INIT: waited 0 seconds for NVRAM to be available

Defaulting CPP : Policer rate for all classes will be set to their defaults


Press RETURN to get started!
```

The following example displays software bootup in bundle mode:

```
switch: boot flash: ie35xx_lite_iosxe.17.17.01.SPA.bin

Attempting to boot from [flash: ie35xx_lite_iosxe.17.17.01.SPA.bin]
Located ie35xx_lite_iosxe.17.17.01.SPA.bin
############################################################################################################################
Warning: ignoring ROMMON var "BOOT_PARAM"

Waiting for 120 seconds for other switches to boot
############################################################################################################################
Switch number is 3

              Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706



Cisco IOS Software, L3 Switch Software (ie35xx_IOSXE), Version 17.17.1, RELEASE SOFTWARE
(fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2025 by Cisco Systems, Inc.
Compiled Tue 30-Oct-25 00:36 by mcpre



Cisco IOS-XE software, Copyright (c) 2005-2025 by cisco Systems, Inc.
All rights reserved.  Certain components of Cisco IOS-XE software are
licensed under the GNU General Public License ("GPL") Version 2.0.  The
```

```
software code licensed under GPL Version 2.0 is free software that comes
with ABSOLUTELY NO WARRANTY.  You can redistribute and/or modify such
GPL code under the terms of GPL Version 2.0.  For more details, see the
documentation or "License Notice" file accompanying the IOS-XE software,
or the applicable URL provided on the flyer accompanying the IOS-XE
software.



FIPS: Flash Key Check : Begin
FIPS: Flash Key Check : End, Not Found, FIPS Mode Not Enabled

This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html

If you require further assistance please contact us by sending email to
export@cisco.com.

cisco IE35xx-25P-4G (ARM64) processor with 518473K/3071K bytes of memory.
Processor board ID JPG221000RH
988 Virtual Ethernet interfaces
56 Gigabit Ethernet interfaces
2048K bytes of non-volatile configuration memory.
2015456K bytes of physical memory.
819200K bytes of Crash Files at crashinfo:.
1941504K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.
819200K bytes of Crash Files at crashinfo-7:.
1941504K bytes of Flash at flash-7:.

Base Ethernet MAC Address          : 68:2c:7b:f7:49:00
Motherboard Assembly Number        : 73-18699-2
Motherboard Serial Number          : JAE22090AZB
Model Revision Number              : 13
Motherboard Revision Number        : 05
Model Number                       : IE35xx-24P-4G
System Serial Number               : JPG221000RH


%INIT: waited 0 seconds for NVRAM to be available

Defaulting CPP : Policer rate for all classes will be set to their defaults


Press RETURN to get started!
```

Example: Managing an Update Package

The following example shows how to add a software package file:

```
Device# install add file flash:ie35xx-17.17.1.spa.bin activate commit

install_add_activate_commit: START Thu Aug 30 20:25:35 IST 2018

Aug 30 20:25:38.688 IST: %INSTALL-5-INSTALL_START_INFO: Switch 7 R0/0: install_engine:
Started install one-shot flash:ie35xx-17.17.1.spa.bin install_add_activate_commit: Adding
PACKAGE

This operation requires a reload of the system. Do you want to proceed?
Please confirm you have changed boot config to flash:packages.conf [y/n]y

--- Starting initial file syncing ---
[7]: Copying flash:ie35xx-17.17.1.spa.bin from switch 7 to switch 4
[4]: Finished copying to switch 4
Info: Finished copying flash:ie35xx-17.17.1.spa.bin to the selected switch(es)
Finished initial file syncing

--- Starting Add ---
Performing Add on all members
  [4] Add package(s) on switch 4
  [4] Finished Add on switch 4
  [7] Add package(s) on switch 7
  [7] Finished Add on switch 7
Checking status of Add on [4 7]
Add: Passed on [4 7]
Finished Add

install_add_activate_commit: Activating PACKAGE

gzip: initramfs.cpio.gz: decompression OK, trailing garbage ignored
Following packages shall be activated:
/flash/ie35xx-webui.17.17.1.SPA.pkg
/flash/ie35xx-srdriver.17.17.1.SPA.pkg
/flash/ie35xx-rpboot.17.17.1.SPA.pkg
/flash/ie35xx-rpbase.17.17.1.SPA.pkg

This operation requires a reload of the system. Do you want to proceed? [y/n]y
--- Starting Activate ---
Performing Activate on all members

Aug 30 20:51:16.365 IST: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 7 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds  [4] Activate package(s)
 on switch 4
  [4] Finished Activate on switch 4
  [7] Activate package(s) on switch 7

Aug 30 20:51:17.561 IST: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 4 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds  [7] Finished Activate
 on switch 7
Checking status of Activate on [4 7]
Activate: Passed on [4 7]
Finished Activate

--- Starting Commit ---
Performing Commit on all members
  [4] Commit package(s) on switch 4
  [4] Finished Commit on switch 4
  [7] Commit package(s) on switch 7
  [7] Finished Commit on switch 7
Checking status of Commit on [4 7]
Commit: Passed on [4 7]
Finished Commit
```

```
Install will reload the system now!
SUCCESS: install_add_activate_commit  Thu Aug 30 20:51:55 IST 2018

Y2#
   Chassis 7 reloading, reason - Reload command

Aug 30 20:51:56.017 IST: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 7 R0/0: install_engine:
 Completed install one-shot PACKAGE flash:ie35xx-17.17.1.spa.bin Aug 30 20:52:03.517:
%PMAN-5-EXITACTION: F0/0: pvp: Process manager is exiting: reload fp action requested
Aug 30 20:52:07.543: %PMAN-5-EXITACTION: R0/0: pvp: Process manager is exiting: rp processes
 exit with reload switch code

Aug 30 20:52:11.104: %PMAN-5-EXITACTION: C0/0: pvp: Process manager is exiting: reload cc
action requested
reboot: Restarting system
```

The following is a sample output of the **show install summary** command after adding a software package file to a device:

```
Device# show install summary
[ Switch 4 7 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
IMG   C    16.9.1.0.70

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

Verifying Software Install

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **show install log**

Example:

```
Device# show install log
```

Displays information about all the software install operations that was performed since boot-up of the device.

```
Device# show install log
[0|install_op_boot]: START Tue Mar 30 06:39:48 Universal 2025
[0|install_op_boot]: END SUCCESS  Tue Mar 30 06:39:50 Universal 2025
```

Step 3 **show install summary**

Example:

```
Device# show install summary
```

Displays information about the image versions and their corresponding install state for all members/field-replaceable unit (FRU).

- The output of this command differs based on the **install** command that is executed.

```
Device# show install summary
[ Switch 1 2 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
IMG   C    17.17.2.0

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

Step 4 **show install package** *filesystem: filename*

Example:

```
Device# show install package flash:ie35xx_lite-rpboot.17.17.01.SPA.pkg
```

Displays information about the specified software install package file.

```
Device# show install package flash:ie35xx_lite-rpboot.17.17.01.SPA.pkg
Package: ie35xx_lite-rpboot.17.17.01.SPA.pkg
  Size: 34616705
  Timestamp: Thu Mar 30 20:28:25 2025 UTC
  Canonical path: /flash/ie35xx_lite-rpboot.17.17.01.SPA.pkg

  Raw disk-file SHA1sum:
    5e816f97bcae3e30eb8bc2f0ec8f64402cea1638
  Header size:     980 bytes
  Package type:    30001
  Package flags:   0
  Header version:  3

  Package is bootable on RP when specified
  by packages provisioning file.
```

Example: Configuring a Device to Download Configurations from a DHCP Server

The following example shows how to use a Layer 3 SVI interface on VLAN 99 to enable DHCP-based autoconfiguration with a saved configuration:

```
Device# configure terminal
Device(config)# boot host dhcp
Device(config)# boot host retry timeout 300
Device(config)# banner config-save ^C Caution - Saving Configuration File to NVRAM May Cause
```

```
 You to No longer Automatically Download Configuration Files at Reboot^C
Device(config)# vlan 99
Device(config-vlan)# interface vlan 99
Device(config-if)# no shutdown
Device(config-if)# end
Device# show boot

BOOT path-list:
Config file:          flash:/config.text
Private Config file:  flash:/private-config.text
Enable Break:         no
Manual Boot:          no
HELPER path-list:
NVRAM/Config file
      buffer size:    32768
Timeout for Config
          Download:    300 seconds
Config Download
       via DHCP:       enabled (next boot: enabled)
Device#
```

Example: Scheduling Software Image Reload

This example shows how to reload the software on a device on the current day at 7:30 p.m:

```
Device# reload at 19:30

Reload scheduled for 19:30:00 UTC Wed Jun 5 2013 (in 2 hours and 25 minutes)
Proceed with reload? [confirm]
```

This example shows how to reload the software on a device at a future date and time:

```
Device# reload at 02:00 jun 20

Reload scheduled for 02:00:00 UTC Thu Jun 20 2013 (in 344 hours and 53 minutes)
Proceed with reload? [confirm]
```

CHAPTER 137

Configuring Application Visibility and Control in a Wired Network

Information About Application Visibility and Control in a Wired Network

Application Visibility and Control (AVC) is a critical part of Cisco's efforts to evolve its Branch and Campus solutions from being strictly packet and connection based to being application-aware and application-intelligent. Application Visibility and Control (AVC) classifies applications using deep packet inspection techniques with the Network-Based Application Recognition (NBAR2) engine. AVC can be configured on wired access ports for standalone switches. NBAR2 can be activated either explicitly on the interface by enabling protocol-discovery or implicitly by attaching a QoS policy that contains **match protocol** classifier. Wired AVC Flexible NetFlow (FNF) can be configured on an interface to provide client, server and application statistics per interface. The record is similar to **application-client-server-stats** traffic monitor which is available in **application-statistics** and **application-performance** profiles in Easy Performance Monitor (Easy perf-mon or ezPM).

Supported AVC Class Map and Policy Map Formats

This section descriobes the supported avc class maps and policy map formats.

Supported AVC Class Map Format

Class Map Format	Class Map Example	Direction
match protocol *protocol name*	`class-map match-any NBAR-VOICE` `match protocol ms-lync-audio`	Both ingress and egress
Combination filters	`class-map match-any NBAR-VOICE` `match protocol ms-lync-audio` `match dscp ef`	Both ingress and egress

Supported AVC Policy Format

Policy Format	QoS Action
Egress policy based on match protocol filter	Mark and police
Ingress policy based on match protocol filter	Mark and police

The following table describes the detailed AVC policy format with an example:

AVC Policy Format	AVC Policy Example	Direction
Basic set	`policy-map MARKING-IN` `class NBAR-MM_CONFERENCING` `set dscp af41`	Ingress and egress
Basic police	`policy-map POLICING-IN` `class NBAR-MM_CONFERENCING` `police cir 600000` `set dscp af41`	Ingress and egress
Basic set and police	`policy-map webex-policy` ` class webex-class` ` set dscp ef` ` police 5000000`	Ingress and egress
Multiple set and police including default	`policy-map webex-policy` ` class webex-class` ` set dscp af31` ` police 4000000` ` class class-webex-category` ` set dscp ef` `  police 6000000` ` class class-default` ` set dscp <>`	Ingress and egress

AVC Policy Format	AVC Policy Example	Direction
Hierarchical police	`policy-map webex-policy` ` class webex-class` ` police 5000000` ` service-policy client-in-police-only` ` policy-map client-in-police-only` ` class webex-class` ` police 100000` ` class class-webex-category` ` set dscp ef` `  police 200000`	Ingress and egress
Hierarchical set and police	`policy-map webex-policy` ` class class-default` ` police 1500000` ` service policy client-up-child` ` policy-map client-up-child` ` class webex-class` ` police 100000` ` set dscp ef` ` class class-webex-category` ` police 200000` ` set dscp af31`	

Restrictions for Wired Application Visibility and Control

- NBAR based QoS policy configuration is not supported on port-channel member ports and virtual interfaces like SVIs or sub-interfaces.
- NBAR based QoS policy configuration is supported on Layer 2 access and trunk ports and Layer 3 routed ports.
- NBAR and transmit (Tx) Switched Port Analyzer (SPAN) is not supported on the same interface.
- Only one of the NBAR based QoS mechanisms are allowed to be attached to any port at the same time, either protocol based or attributes based. Only the following two attributes are supported :
  - traffic-class
  - business-relevance
- The legacy WDAVC QoS limitations are still applicable:
  - Only marking and policing are supported.
  - Only physical interfaces are supported.
  - There is a delay in the QoS classification since the application classification is done offline (while the initial packet/s of the flow are meanwhile forwarded before the correct QoS classification).
- NBAR2 based match criteria **match protocol** will be allowed only with marking or policing actions. NBAR2 match criteria will not be allowed in a policy that has queuing features configured.

- 'Match Protocol': up to 255 concurrent different protocols in all policies (8 bits HW limitation).
- IPv6 packet classification is not supported.
- Only IPv4 unicast(TCP/UDP) is supported.
- Only IPv4 unicast(TCP/UDP) is supported when 'match application name' is used in Netflow record.
- Web UI: You can configure application visibility and perform application monitoring from the Web UI. Application Control can only be done using the CLI. It is not supported on the Web UI.

  To manage and check wired AVC traffic on the Web UI, you must first configure **ip http authentication local** and **ip nbar http-service** commands using the CLI.
- NBAR and ACL logging cannot be configured together on the same switch.
- Protocol-discovery, application-based QoS, and wired AVC FNF cannot be configured together at the same time on the same interface with the non-application-based FNF. However, these wired AVC features can be configured with each other. For example, protocol-discovery, application-based QoS and wired AVC FNF can be configured together on the same interface at the same time.
- Only two wired AVC monitors each with a different predefined record can be attached to an interface at the same time.
- Two directional flow records - ingress and egress - and two legacy flow records are supported.
- Attachment should be done only on physical Layer 2 and Layer 3 ports, and these ports cannot be part of a port channel. Attachment to trunk ports are not supported.
- Scale: Able to handle up to 5000 bi-directional flows per 24 and 48 access ports.
- Wired AVC allows only the fixed set of fields listed in the procedures of this chapter. Other combinations are not allowed. For a regular FNF flow monitor, other combinations are allowed (for the list of supported FNF fields, refer the "Configuring Flexible NetFlow" chapter of the *Network Management Configuration Guide*).
- A new flow record has been included - the DNS flow record. The DNS flow record is similar to the 5-tuple record and includes the DNS domain name field. It accounts only for DNS related fields. This record doesn't have the interface field as a match filed, so the information from all interfaces is aggregated into the same record.

How to Configure Application Visibility and Control

Configuring Application Visibility and Control in a Wired Network

To configure application visibility and control on wired ports, follow these steps:

Configuring Visibility :

- Activate NBAR2 engine by enabling protocol-discovery on the interface using the **ip nbar protocol-discovery** command in the interface configuration mode. See the section, "Enabling Application Recognition on an Interface."

Configuring Control : Configure QoS policies based on application by

1. Creating an AVC QoS policy. See the section, "Creating AVC QoS Policy".
2. Applying AVC QoS policy to the interface. See the section, "Applying a QoS Policy to the Switch Port".

Configuring application-based Flexible Netflow :

- Create a flow record by specifying key and non-key fields to the flow.
- Create a flow exporter to export the flow record.
- Create a flow monitor based on the flow record and the flow exporter.
- Attach the flow monitor to the interface.

Protocol-Discovery, application-based QoS and application-based FNF are all independent features. They can be configured independently or together on the same interface at the same time.

Enabling Application Recognition on an interface

To enable application recognition on an interface, follow these steps:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface for which you are enabling protocol-discovery and enters interface configuration mode.
Step 3	**ip nbar protocol-discovery** **Example:** `Device(config-if)# ip nbar protocol-discovery`	Enables application recognition on the interface by activating NBAR2 engine.
Step 4	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Creating AVC QoS Policy

To create AVC QoS policy, perform these general steps:

1. Create a class map with match protocol filters.
2. Create a policy map.
3. Apply the policy map to the interface.

Creating a Class Map

You need to create a class map before configuring any match protocol filter. The QoS actions such as marking and policing can be applied to the traffic. The AVC match protocol filters are applied to the wired access ports. For more information about the protocols that are supported, see http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/qos_nbar/prot_lib/config_library/nbar-prot-pack-library.html.

Procedure

	Command or Action	Purpose
Step 1	**terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**class-map** *class-map-name* **Example:** `Device(config)# class-map webex-class`	Creates a class map.
Step 3	**match protocol** *application-name* **Example:** `Device(config)# class-map webex-class` `Device(config-cmap)# match protocol webex-media`	Specifies match to the application name.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

Creating a Policy Map

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map webex-policy`	Creates a policy map by entering the policy map name, and enters policy-map configuration mode. By default, no policy maps are defined. The default behavior of a policy map is to set the DSCP to 0 if the packet is an IP packet and to set the CoS to 0 if the packet is tagged. No policing is performed. **Note** To delete an existing policy map, use the **no policy-map** *policy-map-name* global configuration command.
Step 3	**class** [*class-map-name* | **class-default**] **Example:** `Device(config-pmap)# class webex-class`	Defines a traffic classification, and enters policy-map class configuration mode. By default, no policy map and class maps are defined. If a traffic class has already been defined by using the **class-map** global configuration command, specify its name for *class-map-name* in this command. A **class-default** traffic class is predefined and can be added to any policy. It is always placed at the end of a policy map. With an implied **match any** is included in the **class-default** class, all packets that have not already matched the other traffic classes will match **class-default**. **Note** To delete an existing class map, use the **no class** *class-map-name* policy-map configuration command.
Step 4	**police** *rate-bps burst-byte* **Example:** `Device(config-pmap-c)# police 100000 80000`	Defines a policer for the classified traffic. By default, no policer is defined. • For *rate-bps,* specify an average traffic rate in bits per second (b/s). The range is 8000 to 10000000000. • For *burst-byte,* specify the normal burst size in bytes. The range is 1000 to 512000000.

	Command or Action	Purpose
Step 5	**set** {**dscp** *new-dscp* | **cos** *cos-value*} **Example:** `Device(config-pmap-c)# set dscp 45`	Classifies IP traffic by setting a new value in the packet. • For **dscp** *new-dscp*, enter a new DSCP value to be assigned to the classified traffic. The range is 0 to 63.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

Applying a QoS Policy to the switch port

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface Gigabitethernet 1/1`	Enters the interface configuration mode.
Step 3	**service-policy input** *policymapname* **Example:** `Device(config-if)# service-policy input MARKING_IN`	Applies local policy to interface.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

Configuring Wired AVC Flexible Netflow

Creating a Flow Record

Wired AVC FNF supports two types of predefined flow records — Legacy Bidirectional flow records and Directional flow records (ingress and egress). A total of four different predefined flow records, two bidirectional flow records and two directional flow records, can be configured and associated with a flow monitor. The legacy bidirectional records are client/server application statistics records, and the new directional records are application-stats for input/output.

Bidirectional Flow Records

Flow Record 1 - Bidirectional Flow Record

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**flow record** *flow_record_name* **Example:** `Device(config)# flow record fr-wdavc-1`	Enters flow record configuration mode.
Step 3	**description** *description* **Example:** `Device(config-flow-record)# description fr-wdavc-1`	(Optional) Creates a description for the flow record.
Step 4	**match ipv4 version** **Example:** `Device(config-flow-record)# match ipv4 version`	Specifies a match to the IP version from the IPv4 header.
Step 5	**match ipv4 protocol** **Example:** `DEvice(config-flow-record)# match ipv4 protocol`	Specifies a match to the IPv4 protocol.
Step 6	**match application name** **Example:** `Device(config-flow-record)# match application name`	Specifies a match to the application name. **Note** This action is mandatory for AVC support, as this allows the flow to be matched against the application.
Step 7	**match connection client ipv4 address** **Example:** `Device(config-flow-record)# match connection client ipv4 address`	Specifies a match to the IPv4 address of the client (flow initiator).
Step 8	**match connection server ipv4 address** **Example:** `Device(config-flow-record)# match connection server ipv4 address`	Specifies a match to the IPv4 address of the server (flow responder).
Step 9	**match connection server transport port** **Example:**	Specifies a match to the transport port of the server.

	Command or Action	Purpose
	`Device(config-flow-record)# match connection server transport port`	
Step 10	**match flow observation point** **Example:** `Device(config-flow-record)# match flow observation point`	Specifies a match to the observation point ID for flow observation metrics.
Step 11	**collect flow direction** **Example:** `Device(config-flow-record)# collect flow direction`	Specifies to collect the direction — Ingress or Egress — of the relevant side — Initiator or Responder — of the bi-directional flow that is specified by the **initiator** keyword in the **collect connection initiator** command in the step below. Depending on the value specified by the **initiator** keyword, the **flow direction** keyword takes the following values : • 0x01 = Ingress Flow • 0x02 = Egress Flow When the **initiator** keyword is set to initiator, the flow direction is specified from the initiator side of the flow. When the initiator keyword is set to responder, the flow direction is specified from the responder side of the flow. For wired AVC, the **initiator** keyword is always set to initiator.
Step 12	**collect connection initiator** **Example:** `Device(config-flow-record)# collect connection initiator`	Specifies to collect the side of the flow — Initiator or Responder — relevant to the direction of the flow specified by the **collect flow direction** command. The **initiator** keyword provides the following information about the direction of the flow : • 0x01 = Initiator - the flow source is the initiator of the connection For wired AVC, the **initiator** keyword is always set to initiator.
Step 13	**collect connection new-connections** **Example:** `Device(config-flow-record)# collect connection new-connections`	Specifies to collect the number of connection initiations observed.
Step 14	**collect connection client counter packets long** **Example:**	Specifies to collect the number of packets sent by the client.

	Command or Action	Purpose
	`Device(config-flow-record)# collect connection client counter packets long`	
Step 15	**collect connection client counter bytes network long** **Example:** `Device(config-flow-record)# collect connection client counter bytes network long`	Specifies to collect the total number of bytes transmitted by the client.
Step 16	**collect connection server counter packets long** **Example:** `Device(config-flow-record)# collect connection server counter packets long`	Specifies to collect the number of packets sent by the server.
Step 17	**collect connection server counter bytes network long** **Example:** `Device(config-flow-record)# collect connection server counter bytes network long`	Specifies to collect the total number of bytes transmitted by the server.
Step 18	**collect timestamp absolute first** **Example:** `Device(config-flow-record)# collect timestamp absolute first`	Specifies to collect the time, in milliseconds, when the first packet was seen in the flow.
Step 19	**collect timestamp absolute last** **Example:** `Device(config-flow-record)# collect timestamp absolute last`	Specifies to collect the time, in milliseconds, when the most recent packet was seen in the flow.
Step 20	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.
Step 21	**show flow record** **Example:** `Device# show flow record`	Displays information about all the flow records.

Flow Record 2 - Bidirectional Flow Record

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**flow record** *flow_record_name* **Example:** `Device(config)# flow record fr-wdavc-1`	Enters flow record configuration mode.
Step 3	**description** *description* **Example:** `Device(config-flow-record)# description fr-wdavc-1`	(Optional) Creates a description for the flow record.
Step 4	**match ipv4 version** **Example:** `Device(config-flow-record)# match ipv4 version`	Specifies a match to the IP version from the IPv4 header.
Step 5	**match ipv4 protocol** **Example:** `Device(config-flow-record)# match ipv4 protocol`	Specifies a match to the IPv4 protocol.
Step 6	**match application name** **Example:** `Device(config-flow-record)# match application name`	Specifies a match to the application name. **Note** This action is mandatory for AVC support, as this allows the flow to be matched against the application.
Step 7	**match connection client ipv4 address** **Example:** `Device(config-flow-record)# match connection client ipv4 address`	Specifies a match to the IPv4 address of the client (flow initiator).
Step 8	**match connection client transport port** **Example:** `Device(config-flow-record)# match connection client transport port`	(Optional) Specifies a match to the connection port of the client as a key field for a flow record.
Step 9	**match connection server ipv4 address** **Example:** `Device(config-flow-record)# match connection server ipv4 address`	Specifies a match to the IPv4 address of the server (flow responder).

	Command or Action	Purpose
Step 10	**match connection server transport port** **Example:** `Device(config-flow-record)# match connection server transport port`	Specifies a match to the transport port of the server.
Step 11	**match flow observation point** **Example:** `Device(config-flow-record)# match flow observation point`	Specifies a match to the observation point ID for flow observation metrics.
Step 12	**collect flow direction** **Example:** `Device(config-flow-record)# collect flow direction`	Specifies to collect the direction — Ingress or Egress — of the relevant side — Initiator or Responder — of the bi-directional flow that is specified by the **initiator** keyword in the **collect connection initiator** command in the step below. Depending on the value specified by the **initiator** keyword, the **flow direction** keyword takes the following values : • 0x01 = Ingress Flow • 0x02 = Egress Flow When the **initiator** keyword is set to initiator, the flow direction is specified from the initiator side of the flow. When the initiator keyword is set to responder, the flow direction is specified from the responder side of the flow. For wired AVC, the **initiator** keyword is always set to initiator.
Step 13	**collect connection initiator** **Example:** `Device(config-flow-record)# collect connection initiator`	Specifies to collect the side of the flow — Initiator or Responder — relevant to the direction of the flow specified by the **collect flow direction** command. The **initiator** keyword provides the following information about the direction of the flow : • 0x01 = Initiator - the flow source is the initiator of the connection For wired AVC, the **initiator** keyword is always set to initiator.
Step 14	**collect connection new-connections** **Example:** `Device(config-flow-record)# collect connection new-connections`	Specifies to collect the number of connection initiations observed.

	Command or Action	Purpose
Step 15	**collect connection client counter packets long** **Example:** `Device(config-flow-record)# collect connection client counter packets long`	Specifies to collect the number of packets sent by the client.
Step 16	**collect connection client counter bytes network long** **Example:** `Device(config-flow-record)# collect connection client counter bytes network long`	Specifies to collect the total number of bytes transmitted by the client.
Step 17	**collect connection server counter packets long** **Example:** `Device(config-flow-record)# collect connection server counter packets long`	Specifies to collect the number of packets sent by the server.
Step 18	**collect connection server counter bytes network long** **Example:** `Device(config-flow-record)# collect connection server counter bytes network long`	Specifies to collect the total number of bytes transmitted by the server.
Step 19	**collect timestamp absolute first** **Example:** `Device(config-flow-record)# collect timestamp absolute first`	Specifies to collect the time, in milliseconds, when the first packet was seen in the flow.
Step 20	**collect timestamp absolute last** **Example:** `Device(config-flow-record)# collect timestamp absolute last`	Specifies to collect the time, in milliseconds, when the most recent packet was seen in the flow.
Step 21	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.
Step 22	**show flow record** **Example:** `Device# show flow record`	Displays information about all the flow records.

Directional Flow Records

Flow Record 3 - Directional Flow Record - Ingress

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**flow record** *flow_record_name* **Example:** `Device(config)# flow record fr-wdavc-3`	Enters flow record configuration mode.
Step 3	**description** *description* **Example:** `Device(config-flow-record)# description flow-record-1`	(Optional) Creates a description for the flow record.
Step 4	**match ipv4 version** **Example:** `Device(config-flow-record)# match ipv4 version`	Specifies a match to the IP version from the IPv4 header.
Step 5	**match ipv4 protocol** **Example:** `Device(config-flow-record)# match ipv4 protocol`	Specifies a match to the IPv4 protocol.
Step 6	**match ipv4 source address** **Example:** `Device(config-flow-record)# match ipv4 source address`	Specifies a match to the IPv4 source address as a key field.
Step 7	**match ipv4 destination address** **Example:** `Device(config-flow-record)# match ipv4 destination address`	Specifies a match to the IPv4 destination address as a key field.
Step 8	**match transport source-port** **Example:** `Device(config-flow-record)# match transport source-port`	Specifies a match to the transport source port as a key field.
Step 9	**match transport destination-port** **Example:** `Device(config-flow-record)# match transport destination-port`	Specifies a match to the transport destination port as a key field.

	Command or Action	Purpose
Step 10	**match interface input** **Example:** `Device(config-flow-record)# match interface input`	Specifies a match to the input interface as a key field.
Step 11	**match application name** **Example:** `Device(config-flow-record)# match application name`	Specifies a match to the application name. **Note** This action is mandatory for AVC support, as this allows the flow to be matched against the application.
Step 12	**collect interface output** **Example:** `Device(config-flow-record)# collect interface output`	Specifies to collect the output interface from the flows.
Step 13	**collect counter bytes long** **Example:** `Device(config-flow-record)# collect counter bytes long`	Specifies to collect the number of bytes in a flow.
Step 14	**collect counter packets long** **Example:** `Device(config-flow-record)# collect counter packets long`	Specifies to collect the number of packets in a flow.
Step 15	**collect timestamp absolute first** **Example:** `Device(config-flow-record)# collect timestamp absolute first`	Specifies to collect the time, in milliseconds, when the first packet was seen in the flow.
Step 16	**collect timestamp absolute last** **Example:** `Device(config-flow-record)# collect timestamp absolute last`	Specifies to collect the time, in milliseconds, when the most recent packet was seen in the flow.
Step 17	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.
Step 18	**show flow record** **Example:** `Device# show flow record`	Displays information about all the flow records.

Flow Record 4 - Directional Flow Record - Egress

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**flow record** *flow_record_name* **Example:** `Device(config)# flow record fr-wdavc-4`	Enters flow record configuration mode.
Step 3	**description** *description* **Example:** `Device(config-flow-record)# description flow-record-1`	(Optional) Creates a description for the flow record.
Step 4	**match ipv4 version** **Example:** `Device(config-flow-record)# match ipv4 version`	Specifies a match to the IP version from the IPv4 header.
Step 5	**match ipv4 protocol** **Example:** `Device(config-flow-record)# match ipv4 protocol`	Specifies a match to the IPv4 protocol.
Step 6	**match ipv4 source address** **Example:** `Device(config-flow-record)# match ipv4 source address`	Specifies a match to the IPv4 source address as a key field.
Step 7	**match ipv4 destination address** **Example:** `Device(config-flow-record)# match ipv4 destination address`	Specifies a match to the IPv4 destination address as a key field.
Step 8	**match transport source-port** **Example:** `Device(config-flow-record)# match transport source-port`	Specifies a match to the transport source port as a key field.
Step 9	**match transport destination-port** **Example:** `Device(config-flow-record)# match transport destination-port`	Specifies a match to the transport destination port as a key field.

	Command or Action	Purpose
Step 10	**match interface output** **Example:** `Device(config-flow-record)# match interface output`	Specifies a match to the output interface as a key field.
Step 11	**match application name** **Example:** `Device(config-flow-record)# match application name`	Specifies a match to the application name. **Note** This action is mandatory for AVC support, as this allows the flow to be matched against the application.
Step 12	**collect interface input** **Example:** `Device(config-flow-record)# collect interface input`	Specifies to collect the input interface from the flows.
Step 13	**collect counter bytes long** **Example:** `Device(config-flow-record)# collect counter bytes long`	Specifies to collect the number of bytes in a flow.
Step 14	**collect counter packets long** **Example:** `Device(config-flow-record)# collect counter packets long`	Specifies to collect the number of packets in a flow.
Step 15	**collect timestamp absolute first** **Example:** `Device(config-flow-record)# collect timestamp absolute first`	Specifies to collect the time, in milliseconds, when the first packet was seen in the flow.
Step 16	**collect timestamp absolute last** **Example:** `Device(config-flow-record)# collect timestamp absolute last`	Specifies to collect the time, in milliseconds, when the most recent packet was seen in the flow.
Step 17	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.
Step 18	**show flow record** **Example:** `Device# show flow record`	Displays information about all the flow records.

DNS Flow Record

Flow Record 5 - DNS Flow Record

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**flow record** *flow_record_name* **Example:** `Device(config)# flow record fr-wdavc-5`	Enters flow record configuration mode.
Step 3	**description** *description* **Example:** `Device(config-flow-record)# description flow-record-5`	(Optional) Creates a description for the flow record.
Step 4	**match ipv4 version** **Example:** `Device(config-flow-record)# match ipv4 version`	Specifies a match to the IP version from the IPv4 header.
Step 5	**match ipv4 protocol** **Example:** `Device(config-flow-record)# match ipv4 protocol`	Specifies a match to the IPv4 protocol.
Step 6	**match application name** **Example:** `Device(config-flow-record)# match application name`	Specifies a match to the application name. **Note** This action is mandatory for AVC support, as this allows the flow to be matched against the application.
Step 7	**match connection client ipv4 address** **Example:** `Device(config-flow-record)# match connection client ipv4 address`	Specifies a match to the IPv4 address of the client (flow initiator).
Step 8	**match connection client transport port** **Example:** `Device(config-flow-record)# match connection client transport port`	Specifies a match to the connection port of the client as a key field for a flow record.
Step 9	**match connection server ipv4 address** **Example:**	Specifies a match to the IPv4 address of the server (flow responder).

	Command or Action	Purpose
	`Device(config-flow-record)# match connection server ipv4 address`	
Step 10	**match connection server transport port** **Example:** `Device(config-flow-record)# match connection server transport port`	Specifies a match to the transport port of the server.
Step 11	**collect flow direction** **Example:** `Device(config-flow-record)# collect flow direction`	Specifies to collect the direction — Ingress or Egress — of the relevant side — Initiator or Responder — of the bi-directional flow that is specified by the **initiator** keyword in the **collect connection initiator** command in the step below. Depending on the value specified by the **initiator** keyword, the **flow direction** keyword takes the following values : • 0x01 = Ingress Flow • 0x02 = Egress Flow When the **initiator** keyword is set to initiator, the flow direction is specified from the initiator side of the flow. When the initiator keyword is set to responder, the flow direction is specified from the responder side of the flow. For wired AVC, the **initiator** keyword is always set to initiator.
Step 12	**collect timestamp absolute first** **Example:** `Device(config-flow-record)# collect timestamp absolute first`	Specifies to collect the time, in milliseconds, when the first packet was seen in the flow.
Step 13	**collect timestamp absolute last** **Example:** `Device(config-flow-record)# collect timestamp absolute last`	Specifies to collect the time, in milliseconds, when the most recent packet was seen in the flow.
Step 14	**collect connection initiator** **Example:** `Device(config-flow-record)# collect connection initiator`	Specifies to collect the side of the flow — Initiator or Responder — relevant to the direction of the flow specified by the **collect flow direction** command. The **initiator** keyword provides the following information about the direction of the flow : • 0x01 = Initiator - the flow source is the initiator of the connection For wired AVC, the **initiator** keyword is always set to initiator.

	Command or Action	Purpose
Step 15	**collect connection new-connections** **Example:** `Device(config-flow-record)# collect connection new-connections`	Specifies to collect the number of connection initiations observed.
Step 16	**collect connection server counter packets long** **Example:** `Device(config-flow-record)# collect connection server counter packets long`	Specifies to collect the number of packets sent by the server.
Step 17	**collect connection client counter packets long** **Example:** `Device(config-flow-record)# collect connection client counter packets long`	Specifies to collect the number of packets sent by the client.
Step 18	**collect connection server counter bytes network long** **Example:** `Device(config-flow-record)# collect connection server counter bytes network long`	Specifies to collect the total number of bytes transmitted by the server.
Step 19	**collect connection client counter bytes network long** **Example:** `Device(config-flow-record)# collect connection client counter bytes network long`	Specifies to collect the total number of bytes transmitted by the client.
Step 20	**collect application dns domain-name** **Example:** `Device(config-flow-record)# collect application dns domain-name`	Configures the use of the DNS Domain-Name as a Collect field for a DNS flow record.
Step 21	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

Creating a Flow Exporter

You can create a flow exporter to define the export parameters for a flow.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**flow exporter** *flow_exporter_name* **Example:** `Device(config)# flow exporter flow-exporter-1`	Enters flow exporter configuration mode.
Step 3	**description** *description* **Example:** `Device(config-flow-exporter)# description flow-exporter-1`	(Optional) Creates a description for the flow exporter.
Step 4	**destination** { *hostname* | *ipv4-address* | *ipv6-address* } **Example:** `Device(config-flow-exporter)# destination 10.10.1.1`	Specifies the hostname, IPv4 or IPv6 address of the system to which the exporter sends data.
Step 5	**option application-table** [**timeout** *seconds*] **Example:** `Device(config-flow-exporter)# option application-table timeout 500`	(Optional) Configures the application table option for the flow exporter. The **timeout** option configures the resend time in seconds for the flow exporter. The valid range is from 1 to 86400 seconds.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.
Step 7	**show flow exporter** **Example:** `Device# show flow exporter`	Displays information about all the flow exporters.
Step 8	**show flow exporter statistics** **Example:** `Device# show flow exporter statistics`	Displays flow exporter statistics.

Creating a Flow Monitor

You can create a flow monitor and associate it with a flow record.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**flow monitor** *monitor-name* **Example:** `Device(config)# flow monitor flow-monitor-1`	Creates a flow monitor and enters flow monitor configuration mode.
Step 3	**description** *description* **Example:** `Device(config-flow-monitor)# description flow-monitor-1`	(Optional) Creates a description for the flow monitor.
Step 4	**record** *record-name* **Example:** `Device(config-flow-monitor)# record flow-record-1`	Specifies the name of a record that was created previously.
Step 5	**exporter** *exporter-name* **Example:** `Device(config-flow-monitor)# exporter flow-exporter-1`	Specifies the name of an exporter that was created previously.
Step 6	**cache** { **entries** *number-of-entries* | **timeout** {**active** | **inactive**} | **type normal** } **Example:** `Device(config-flow-monitor)# cache timeout active 1800` **Example:** `Device(config-flow-monitor)# cache timeout inactive 200` **Example:** `Device(config-flow-monitor)# cache type normal`	(Optional) Specifies to configure flow cache parameters. • **entries** *number-of-entries* — Specifies the maximum number of flow entries in the flow cache in the range from 16 to 65536. **Note** Only normal cache type is supported.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.
Step 8	**show flow monitor** **Example:**	Displays information about all the flow monitors.

	Command or Action	Purpose
	`Device# show flow monitor`	
Step 9	**show flow monitor** *flow-monitor-name* **Example:** `Device# show flow monitor flow-monitor-1`	Displays information about the specified wired AVC flow monitor.
Step 10	**show flow monitor** *flow-monitor-name* **statistics** **Example:** `Device# show flow monitor flow-monitor-1 statistics`	Displays statistics for wired AVC flow monitor.
Step 11	**clear flow monitor** *flow-monitor-name* **statistics** **Example:** `Device# clear flow monitor flow-monitor-1 statistics`	Clears the statistics of the specified flow monitor. Use the **show flow monitor flow-monitor-1 statistics** command after using the **clear flow monitor flow-monitor-1 statistics** to verify that all the statistics have been reset.
Step 12	**show flow monitor** *flow-monitor-name* **cache format table** **Example:** `Device# show flow monitor flow-monitor-1 cache format table`	Displays flow cache contents in a tabular format.
Step 13	**show flow monitor** *flow-monitor-name* **cache format record** **Example:** `Device# show flow monitor flow-monitor-1 cache format record`	Displays flow cache contents in similar format as the flow record.
Step 14	**show flow monitor** *flow-monitor-name* **cache format csv** **Example:** `Device# show flow monitor flow-monitor-1 cache format csv`	Displays flow cache contents in CSV format.

Associating Flow Monitor to an interface

You can attach two different wired AVC monitors with different predefined records to an interface at the same time.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface Gigabitethernet 1/1`	Enters the interface configuration mode.
Step 3	**ip flow monitor** *monitor-name* { **input** | **output** } **Example:** `Device(config-if) # ip flow monitor flow-monitor-1 input`	Associates a flow monitor to the interface for input and/or output packets.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

NBAR2 Custom Applications

NBAR2 supports the use of custom protocols to identify custom applications. Custom protocols support protocols and applications that NBAR2 does not currently support.

In every deployment, there are local and specific applications which are not covered by the NBAR2 protocol pack provided by Cisco. Local applications are mainly categorized as:

- Specific applications to an organization
- Applications specific to a geography

NBAR2 provides a way to manually customize such local applications. You can manually customize applications using the command **ip nbar custom** *myappname* in global configuration mode. Custom applications take precedence over built-in protocols. For each custom protocol, user can define a selector ID that can be used for reporting purposes.

There are various types of application customization:

Generic protocol customization

- HTTP
- SSL
- DNS

Composite : Customization based on multiple underlying protocols – **server-name**

Layer3/Layer4 customization

- IPv4 address
- DSCP values
- TCP/UDP ports
- Flow source or destination direction

Byte Offset : Customization based on specific byte values in the payload

HTTP Customization

HTTP customization could be based on a combination of HTTP fields from:

- **cookie** - HTTP Cookie
- **host** - Host name of Origin Server containing resource
- **method** - HTTP method
- **referrer** - Address the resource request was obtained from
- **url** - Uniform Resource Locator path
- **user-agent** - Software used by agent sending the request
- **version** - HTTP version
- **via** - HTTP via field

HTTP Customization

Custom application called MYHTTP using the HTTP host "*mydomain.com" with Selector ID 10.

```
Device# configure terminal
Device(config)# ip nbar custom MYHTTP http host *mydomain.com id 10
```

SSL Customization

Customization can be done for SSL encrypted traffic using information extracted from the SSL Server Name Indication (SNI) or Common Name (CN).

SSL Customization

Custom application called MYSSL using SSL unique-name "mydomain.com" with selector ID 11.

```
Device# configure terminal
Device(config)#ip nbar custom MYSSL ssl unique-name *mydomain.com id 11
```

DNS Customization

NBAR2 examines DNS request and response traffic, and can correlate the DNS response to an application. The IP address returned from the DNS response is cached and used for later packet flows associated with that specific application.

The command **ip nbar custom** *application-name* **dns** *domain-name* **id** *application-id* is used for DNS customization. To extend an existing application, use the command **ip nbar custom** *application-name* **dns domain-name** *domain-name* **extends** *existing-application*.

For more information on DNS based customization, see http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/qos_nbar/configuration/xe-3s/asr1000/qos-nbar-xe-3s-asr-1000-book/nbar-custapp-dns-xe.html.

DNS Customization

Custom application called MYDNS using the DNS domain name “mydomain.com” with selector ID 12.

```
Device# configure terminal
Device(config)# ip nbar custom MYDNS dns domain-name *mydomain.com id 12
```

Composite Customization

NBAR2 provides a way to customize applications based on domain names appearing in HTTP, SSL or DNS.

Composite Customization

Custom application called MYDOMAIN using HTTP, SSL or DNS domain name “mydomain.com” with selector ID 13.

```
Device# configure terminal
Device(config)# ip nbar custom MYDOMAIN composite server-name *mydomain.com id 13
```

L3/L4 Customization

Layer3/Layer4 customization is based on the packet tuple and is always matched on the first packet of a flow.

L3/L4 Customization

Custom application called LAYER4CUSTOM matching IP addresses 10.56.1.10 and 10.56.1.11, TCP and DSCP ef with selector ID 14.

```
Device# configure terminal
Device(config)# ip nbar custom LAYER4CUSTOM transport tcp id 14
Device(config-custom)# ip address 10.56.1.10 10.56.1.11
Device(config-custom)# dscp ef
```

Examples: Monitoring Custom Applications

Show Commands for Monitoring Custom Applications

show ip nbar protocol-id | inc Custom

```
Device# show ip nbar  protocol-id | inc Custom
LAYER4CUSTOM              14              Custom
MYDNS                     12              Custom
MYDOMAIN                  13              Custom
MYHTTP                    10              Custom
MYSSL                     11              Custom
```

show ip nbar protocol-discovery protocol *CUSTOM_APP*

```
Device# show ip nbar  protocol-id MYSSL
Protocol Name               id             type
-----------------------------------------------
MYSSL                      11             Custom
```

NBAR2 Dynamic Hitless Protocol Pack Upgrade

Protocol packs are software packages that update the NBAR2 protocol support on a device without replacing the Cisco software on the device. A protocol pack contains information on applications officially supported by NBAR2 which are compiled and packed together. For each application, the protocol-pack includes information on application signatures and application attributes. Each software release has a built-in protocol-pack bundled with it.

Protocol packs provide the following features:

- They are easy and fast to load.
- They are easy to upgrade to a higher version protocol pack or revert to a lower version protocol pack.
- They do not require the switch to be reloaded.

NBAR2 protocol packs are available for download on Cisco Software Center from this URL: https://software.cisco.com/download/home .

Prerequisites for the NBAR2 Protocol Pack

Before loading a new protocol pack, you must copy the protocol pack to the flash on all the switch members.

To load a protocol pack, see Loading the NBAR2 Protocol Pack, on page 2068 .

Loading the NBAR2 Protocol Pack

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar protocol-pack** *protocol-pack* [**force**] **Example:** `Device(config)# ip nbar protocol-pack flash:defProtoPack` **Example:**	Loads the protocol pack. • Use the **force** keyword to specify and load a protocol pack of a lower version, which is different from the base protocol pack version. This also removes the configuration that is not supported by the current protocol pack on the switch.

	Command or Action	Purpose
	`Device(config)# default ip nbar protocol-pack`	For reverting to the built-in protocol pack, use the following command:
Step 4	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 5	**show ip nbar protocol-pack** {protocol-pack | **active**} [**detail**] **Example:** `Device# show ip nbar protocol-pack active`	Displays the protocol pack information. • Verify the loaded protocol pack version, publisher, and other details using this command. • Use the *protocol-pack* argument to display information about the specified protocol pack. • Use the **active** keyword to display active protocol pack information. • Use the **detail** keyword to display detailed protocol pack information.

Examples: Loading the NBAR2 Protocol Pack

The following example shows how to load a new protocol pack:

```
Device> enable
Device# configure terminal
Device(config)# ip nbar protocol-pack flash:newDefProtoPack
Device(config)# exit
```

The following example shows how to use the **force** keyword to load a protocol pack of a lower version:

```
Device> enable
Device# configure terminal
Device(config)# ip nbar protocol-pack flash:OldDefProtoPack force
Device(config)# exit
```

The following example shows how to revert to the built-in protocol pack:

```
Device> enable
Device# configure terminal
Device(config)# default ip nbar protocol-pack
Device(config)# exit
```

Monitoring Application Visibility and Control

This section describes the new commands for application visibility.

The following commands can be used to monitor application visibility on the switch and access ports.

Table 143: Monitoring Application Visibility Commands on the Switch

Command	Purpose
show ip nbar protocol-discovery [**interface** *interface-type interface-number*] [**stats**{**byte-count** | **bit-rate** | **packet-count** | **max-bit-rate**}] [**protocol** *protocol-name* | **top-n** *number*]	Displays the statistics gathered by the NBAR Protocol Discovery feature. • (Optional) Enter keywords and arguments to fine-tune the statistics displayed. For more information on each of the keywords, refer to the **show ip nbar protocol-discovery**command in Cisco IOS Quality of Service Solutions Command Reference.
show policy-map interface *interface-type interface-number*	Displays information about policy map applied to the interface.

Examples: Application Visibility and Control Configuration

This example shows how to create class maps with apply match protocol filters for application name:

```
Device# configure terminal
Device(config)# class-map match-any NBAR-VOICE
Device(config-cmap)# match protocol ms-lync-audio
Device(config-cmap)#end
```

This example shows how to create policy maps and define existing class maps for egress QoS:

```
Device # configure terminal
Device(config)# policy-map test-avc-up
Device(config-pmap)# class cat-browsing
Device(config-pmap-c)# police 150000
Device(config-pmap-c)# set dscp 12
Device(config-pmap-c)#end
```

This example shows how to create policy maps and define existing class maps for ingress QoS:

```
Device# configure terminal
Device(config)# policy-map test-avc-down
Device(config-pmap)# class cat-browsing
Device(config-pmap-c)# police 200000
Device(config-pmap-c)# set dscp 10
Device(config-pmap-c)#end
```

This example shows how to apply policy maps to a switch port:

```
Device# configure terminal
Device(config)# interface GigabitEthernet 1/1
Device(config-if)# switchport mode access
Device(config-if)# switchport access vlan 20
Device(config-if)# service-policy input POLICING_IN
Device(config-if)#end
```

This example shows how to create class maps based on NBAR attributes.

```
Device# configure terminal
Device(config)# class-map match-all rel-relevant
Device(config-cmap)# match protocol attribute business-relevance business-relevant

Device(config)# class-map match-all rel-irrelevant
```

```
Device(config-cmap)# match protocol attribute business-relevance business-irrelevant

Device(config)# class-map match-all rel-default
Device(config-cmap)# match protocol attribute business-relevance default

Device(config)# class-map match-all class--ops-admin-and-rel
Device(config-cmap)# match protocol attribute traffic-class ops-admin-mgmt
Device(config-cmap)# match protocol attribute business-relevance business-relevant
```

This example shows how to create policy maps based on class maps based on NBAR attributes.

```
Device# configure terminal
Device(config)# policy-map attrib--rel-types
Device(config-pmap)# class rel-relevant
Device(config-pmap-c)# set dscp ef
Device(config-pmap-c)# class rel-irrelevant
Device(config-pmap-c)# set dscp af11
Device(config-pmap-c)# class rel-default
Device(config-pmap-c)# set dscp default

Device(config)# policy-map attrib--ops-admin-and-rel
Device(config-pmap)# class class--ops-admin-and-rel
Device(config-pmap-c)# set dscp cs5
```

This example shows how to attach a policy map based on NBAR attributes to a wired port:

```
Device# configure terminal
Device(config)# interface GigabitEthernet1/1
Device(config-if)# service-policy input attrib--rel-types
```

Show Commands for Viewing the Configuration

show ip nbar protocol-discovery

Displays a report of the Protocol Discovery statistics per interface.

The following is a sample output for the statistics per interface:

```
Device# show ip nbar protocol-discovery int GigabitEthernet1/1


GigabitEthernet1/1
Last clearing of "show ip nbar protocol-discovery" counters 00:03:16

                                     Input
Output
                                     -----
------
Protocol                             Packet Count
Packet Count
                                     Byte Count
Byte Count
                                     30sec Bit Rate (bps)
30sec Bit Rate (bps)
                                     30sec Max Bit Rate (bps)
30sec Max Bit Rate (bps)
------------------------ ------------------------
-------------------------------------------------
ms-lync                              60580
```

```
55911
                                    31174777
28774864
                                    3613000
93000
                                    3613000
3437000
Total                               60580
55911
                                    31174777
28774864
                                    3613000
93000
                                    3613000
3437000
```

show policy-map interface

Displays the QoS statistics and the configured policy maps on all interfaces.

The following is a sample output for the policy-maps configured on all the interfaces:

```
Device# show policy-map int


GigabitEthernet1/1
 Service-policy input: MARKING-IN

   Class-map: NBAR-VOICE (match-any)
     718 packets
     Match: protocol ms-lync-audio
       0 packets, 0 bytes
       30 second rate 0 bps
     QoS Set
       dscp ef

   Class-map: NBAR-MM_CONFERENCING (match-any)
     6451 packets
     Match: protocol ms-lync
       0 packets, 0 bytes
       30 second rate 0 bps
     Match: protocol ms-lync-video
       0 packets, 0 bytes
       30 second rate 0 bps
     QoS Set
       dscp af41

   Class-map: class-default (match-any)
     34 packets
     Match: any
```

Show Commands for Viewing Attributes-based QoS Configuration

show policy-map interface

Displays the attribute-based QoS statistics and the configured policy maps on all interfaces.

The following is a sample output for the policy-maps configured on all the interfaces:

```
Device# show policy-map interface gigabitEthernet 1/1
 GigabitEthernet1/1

  Service-policy input: attrib--rel-types

    Class-map: rel-relevant (match-all)
      20 packets
      Match: protocol attribute business-relevance business-relevant
      QoS Set
        dscp ef

    Class-map: rel-irrelevant (match-all)
      0 packets
      Match: protocol attribute business-relevance business-irrelevant
      QoS Set
        dscp af11

    Class-map: rel-default (match-all)
      14 packets
      Match: protocol attribute business-relevance default
      QoS Set
        dscp default

    Class-map: class-default (match-any)
      0 packets
      Match: any
```

show ip nbar protocol-attribute

Displays all the protocol attributes used by NBAR.

The following shows sample output for some of the attributes:

```
Device# show ip nbar protocol-attribute cisco-jabber-im
           Protocol Name : cisco-jabber-im
                encrypted : encrypted-yes
                   tunnel : tunnel-no
                 category : voice-and-video
             sub-category : enterprise-media-conferencing
        application-group : cisco-jabber-group
           p2p-technology : p2p-tech-no
            traffic-class : transactional-data
       business-relevance : business-relevant
          application-set : collaboration-apps

Device# show ip nbar protocol-attribute google-services
            Protocol Name : google-services
                encrypted : encrypted-yes
                   tunnel : tunnel-no
                 category : other
             sub-category : other
```

```
       application-group : google-group
          p2p-technology : p2p-tech-yes
           traffic-class : transactional-data
      business-relevance : default
         application-set : general-browsing
```

```
Device# show ip nbar protocol-attribute dns
            Protocol Name : google-services
                encrypted : encrypted-yes
                   tunnel : tunnel-no
                 category : other
              sub-category : other
       application-group : google-group
          p2p-technology : p2p-tech-yes
           traffic-class : transactional-data
      business-relevance : default
         application-set : general-browsing
```

```
Device# show ip nbar protocol-attribute unknown
            Protocol Name : unknown
                encrypted : encrypted-no
                   tunnel : tunnel-no
                 category : other
              sub-category : other
       application-group : other
          p2p-technology : p2p-tech-no
           traffic-class : bulk-data
      business-relevance : default
         application-set : general-misc
```

Show Commands for Viewing Flow Monitor Configuration

show flow monitor wdavc

Displays information about the specified wired AVC flow monitor.

```
Device # show flow monitor wdavc

Flow Monitor wdavc:
  Description:       User defined
  Flow Record:       wdavc
  Flow Exporter:     wdavc-exp (inactive)
  Cache:
    Type:                 normal (Platform cache)
    Status:               not allocated
    Size:                 12000 entries
    Inactive Timeout:     15 secs
    Active Timeout:       1800 secs
```

show flow monitor wdavc statistics

Displays statistics for wired AVC flow monitor.

```
Device# show flow monitor wdavc statistics
  Cache type:                                 Normal (Platform cache)
  Cache size:                                  12000
```

```
  Current entries:                              13

  Flows added:                                  26
  Flows aged:                                   13
    - Active timeout      (  1800 secs)          1
    - Inactive timeout    (    15 secs)         12
```

clear flow monitor wdavc statistics

Clears the statistics of the specified flow monitor. Use the **show flow monitor wdavc statistics** command after using the **clear flow monitor wdavc statistics** to verify that all the statistics have been reset. The following is a sample output of the **show flow monitor wdavc statistics** command after clearing flow monitor statistics.

```
Device# show flow monitor wdavc statistics
  Cache type:                                 Normal (Platform cache)
  Cache size:                                  12000
  Current entries:                                 0

  Flows added:                                     0
  Flows aged:                                      0
```

Show Commands for Viewing Cache Contents

show flow monitor wdavc cache format table

Displays flow cache contents in a tabular format.

```
Device# show flow monitor wdavc cache format table
  Cache type:                                 Normal (Platform cache)
  Cache size:                                  12000
  Current entries:                                13

  Flows added:                                    26
  Flows aged:                                     13
    - Active timeout      (  1800 secs)          1
    - Inactive timeout    (    15 secs)         12

CONN IPV4 INITIATOR ADDR  CONN IPV4 RESPONDER ADDR  CONN RESPONDER PORT
 FLOW OBSPOINT ID  IP VERSION  IP PROT  APP NAME                    flow
 dirn ...................
------------------------  ------------------------  -------------------
 ----------------  ----------  -------  ---------------------------
---------
64.103.125.147            144.254.71.184                              53
       4294967305          4       17  port dns                   Input
     ...................
64.103.121.103            10.1.1.2                                    67
       4294967305          4       17  layer7 dhcp                Input
     ....contd...........
64.103.125.3              64.103.125.97                               68
       4294967305          4       17  layer7 dhcp                Input
     ...................
10.0.2.6                  157.55.40.149                               443
       4294967305          4        6  layer7 ms-lync             Input
```

```
     .....................
64.103.126.28             66.163.36.139                              443
       4294967305          4        6  layer7 cisco-jabber-im       Input
     ....contd............
64.103.125.2              64.103.125.29                               68
       4294967305          4       17  layer7 dhcp                  Input
     .....................
64.103.125.97             64.103.101.181                              67
       4294967305          4       17  layer7 dhcp                  Input
     .....................
192.168.100.6             10.10.20.1                                5060
       4294967305          4       17  layer7 cisco-jabber-control  Input
     ....contd............
64.103.125.3              64.103.125.29                               68
       4294967305          4       17  layer7 dhcp                  Input
     .....................
10.80.101.18              10.80.101.6                               5060
       4294967305          4        6  layer7 cisco-collab-control  Input
     .....................
10.1.11.4                 66.102.11.99                                80
       4294967305          4        6  layer7 google-services       Input
     ....contd............
64.103.125.2              64.103.125.97                               68
       4294967305          4       17  layer7 dhcp                  Input
     .....................
64.103.125.29             64.103.101.181                              67
       4294967305          4       17  layer7 dhcp                  Input
     .....................
```

show flow monitor wdavc cache format record

Displays flow cache contents in similar format as the flow record.

```
Device# show flow monitor wdavc cache format record
  Cache type:                               Normal (Platform cache)
  Cache size:                                12000
  Current entries:                              13

  Flows added:                                  26
  Flows aged:                                   13
    - Active timeout      (  1800 secs)          1
    - Inactive timeout    (    15 secs)         12

CONNECTION IPV4 INITIATOR ADDRESS:        64.103.125.147
CONNECTION IPV4 RESPONDER ADDRESS:        144.254.71.184
CONNECTION RESPONDER PORT:                53
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              17
APPLICATION NAME:                         port dns
flow direction:                           Input
timestamp abs first:                      08:55:46.917
timestamp abs last:                       08:55:46.917
```

```
connection initiator:                     Initiator
connection count new:                     2
connection server packets counter:        1
connection client packets counter:        1
connection server network bytes counter:  190
connection client network bytes counter:  106

CONNECTION IPV4 INITIATOR ADDRESS:        64.103.121.103
CONNECTION IPV4 RESPONDER ADDRESS:        10.1.1.2
CONNECTION RESPONDER PORT:                67
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              17
APPLICATION NAME:                         layer7 dhcp
flow direction:                           Input
timestamp abs first:                      08:55:47.917
timestamp abs last:                       08:55:47.917
connection initiator:                     Initiator
connection count new:                     1
connection server packets counter:        0
connection client packets counter:        1
connection server network bytes counter:  0
connection client network bytes counter:  350

CONNECTION IPV4 INITIATOR ADDRESS:        64.103.125.3
CONNECTION IPV4 RESPONDER ADDRESS:        64.103.125.97
CONNECTION RESPONDER PORT:                68
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              17
APPLICATION NAME:                         layer7 dhcp
flow direction:                           Input
timestamp abs first:                      08:55:47.917
timestamp abs last:                       08:55:53.917
connection initiator:                     Initiator
connection count new:                     1
connection server packets counter:        0
connection client packets counter:        4
connection server network bytes counter:  0
connection client network bytes counter:  1412

CONNECTION IPV4 INITIATOR ADDRESS:        10.0.2.6
CONNECTION IPV4 RESPONDER ADDRESS:        157.55.40.149
CONNECTION RESPONDER PORT:                443
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              6
APPLICATION NAME:                         layer7 ms-lync
flow direction:                           Input
timestamp abs first:                      08:55:46.917
timestamp abs last:                       08:55:46.917
```

```
connection initiator:                     Initiator
connection count new:                     2
connection server packets counter:        10
connection client packets counter:        14
connection server network bytes counter:  6490
connection client network bytes counter:  1639

CONNECTION IPV4 INITIATOR ADDRESS:        64.103.126.28
CONNECTION IPV4 RESPONDER ADDRESS:        66.163.36.139
CONNECTION RESPONDER PORT:                443
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              6
APPLICATION NAME:                         layer7 cisco-jabber-im
flow direction:                           Input
timestamp abs first:                      08:55:46.917
timestamp abs last:                       08:55:46.917
connection initiator:                     Initiator
connection count new:                     2
connection server packets counter:        12
connection client packets counter:        10
connection server network bytes counter:  5871
connection client network bytes counter:  2088

CONNECTION IPV4 INITIATOR ADDRESS:        64.103.125.2
CONNECTION IPV4 RESPONDER ADDRESS:        64.103.125.29
CONNECTION RESPONDER PORT:                68
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              17
APPLICATION NAME:                         layer7 dhcp
flow direction:                           Input
timestamp abs first:                      08:55:47.917
timestamp abs last:                       08:55:47.917
connection initiator:                     Initiator
connection count new:                     1
connection server packets counter:        0
connection client packets counter:        2
connection server network bytes counter:  0
connection client network bytes counter:  712

CONNECTION IPV4 INITIATOR ADDRESS:        64.103.125.97
CONNECTION IPV4 RESPONDER ADDRESS:        64.103.101.181
CONNECTION RESPONDER PORT:                67
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              17
APPLICATION NAME:                         layer7 dhcp
flow direction:                           Input
timestamp abs first:                      08:55:47.917
timestamp abs last:                       08:55:47.917
```

```
connection initiator:                     Initiator
connection count new:                     1
connection server packets counter:        0
connection client packets counter:        1
connection server network bytes counter:  0
connection client network bytes counter:  350

CONNECTION IPV4 INITIATOR ADDRESS:        192.168.100.6
CONNECTION IPV4 RESPONDER ADDRESS:        10.10.20.1
CONNECTION RESPONDER PORT:                5060
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              17
APPLICATION NAME:                         layer7 cisco-jabber-control
flow direction:                           Input
timestamp abs first:                      08:55:46.917
timestamp abs last:                       08:55:46.917
connection initiator:                     Initiator
connection count new:                     1
connection server packets counter:        0
connection client packets counter:        2
connection server network bytes counter:  0
connection client network bytes counter:  2046

CONNECTION IPV4 INITIATOR ADDRESS:        64.103.125.3
CONNECTION IPV4 RESPONDER ADDRESS:        64.103.125.29
CONNECTION RESPONDER PORT:                68
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              17
APPLICATION NAME:                         layer7 dhcp
flow direction:                           Input
timestamp abs first:                      08:55:47.917
timestamp abs last:                       08:55:47.917
connection initiator:                     Initiator
connection count new:                     1
connection server packets counter:        0
connection client packets counter:        2
connection server network bytes counter:  0
connection client network bytes counter:  712

CONNECTION IPV4 INITIATOR ADDRESS:        10.80.101.18
CONNECTION IPV4 RESPONDER ADDRESS:        10.80.101.6
CONNECTION RESPONDER PORT:                5060
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              6
APPLICATION NAME:                         layer7 cisco-collab-control
flow direction:                           Input
timestamp abs first:                      08:55:46.917
timestamp abs last:                       08:55:47.917
```

```
connection initiator:                     Initiator
connection count new:                     2
connection server packets counter:        23
connection client packets counter:        27
connection server network bytes counter:  12752
connection client network bytes counter:  8773

CONNECTION IPV4 INITIATOR ADDRESS:        10.1.11.4
CONNECTION IPV4 RESPONDER ADDRESS:        66.102.11.99
CONNECTION RESPONDER PORT:                80
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              6
APPLICATION NAME:                         layer7 google-services
flow direction:                           Input
timestamp abs first:                      08:55:46.917
timestamp abs last:                       08:55:46.917
connection initiator:                     Initiator
connection count new:                     2
connection server packets counter:        3
connection client packets counter:        5
connection server network bytes counter:  1733
connection client network bytes counter:  663

CONNECTION IPV4 INITIATOR ADDRESS:        64.103.125.2
CONNECTION IPV4 RESPONDER ADDRESS:        64.103.125.97
CONNECTION RESPONDER PORT:                68
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              17
APPLICATION NAME:                         layer7 dhcp
flow direction:                           Input
timestamp abs first:                      08:55:47.917
timestamp abs last:                       08:55:53.917
connection initiator:                     Initiator
connection count new:                     1
connection server packets counter:        0
connection client packets counter:        4
connection server network bytes counter:  0
connection client network bytes counter:  1412

CONNECTION IPV4 INITIATOR ADDRESS:        64.103.125.29
CONNECTION IPV4 RESPONDER ADDRESS:        64.103.101.181
CONNECTION RESPONDER PORT:                67
FLOW OBSPOINT ID:                         4294967305
IP VERSION:                               4
IP PROTOCOL:                              17
APPLICATION NAME:                         layer7 dhcp
flow direction:                           Input
timestamp abs first:                      08:55:47.917
timestamp abs last:                       08:55:47.917
```

```
connection initiator:                       Initiator
connection count new:                       1
connection server packets counter:          0
connection client packets counter:          1
connection server network bytes counter:    0
connection client network bytes counter:    350
```

show flow monitor wdavc cache format csv

Displays flow cache contents in CSV format.

```
Device# show flow monitor wdavc cache format csv
  Cache type:                                 Normal (Platform cache)
  Cache size:                                  12000
  Current entries:                                13

  Flows added:                                    26
  Flows aged:                                     13
    - Active timeout      (  1800 secs)           1
    - Inactive timeout    (    15 secs)          12

CONN IPV4 INITIATOR ADDR,CONN IPV4 RESPONDER ADDR,CONN RESPONDER PORT,FLOW
 OBSPOINT ID,IP VERSION,IP
PROT,APP NAME,flow dirn,time abs first,time abs last,conn initiator,conn
 count new,conn server packets
cnt,conn client packets cnt,conn server network bytes cnt,conn client
network bytes cnt
64.103.125.147,144.254.71.184,53,4294967305,4,17,port
dns,Input,08:55:46.917,08:55:46.917,Initiator,2,1,1,190,106
64.103.121.103,10.1.1.2,67,4294967305,4,17,layer7
dhcp,Input,08:55:47.917,08:55:47.917,Initiator,1,0,1,0,350
64.103.125.3,64.103.125.97,68,4294967305,4,17,layer7
dhcp,Input,08:55:47.917,08:55:53.917,Initiator,1,0,4,0,1412
10.0.2.6,157.55.40.149,443,4294967305,4,6,layer7 ms-
lync,Input,08:55:46.917,08:55:46.917,Initiator,2,10,14,6490,1639
64.103.126.28,66.163.36.139,443,4294967305,4,6,layer7 cisco-jabber-
im,Input,08:55:46.917,08:55:46.917,Initiator,2,12,10,5871,2088
64.103.125.2,64.103.125.29,68,4294967305,4,17,layer7
dhcp,Input,08:55:47.917,08:55:47.917,Initiator,1,0,2,0,712
64.103.125.97,64.103.101.181,67,4294967305,4,17,layer7
dhcp,Input,08:55:47.917,08:55:47.917,Initiator,1,0,1,0,350
192.168.100.6,10.10.20.1,5060,4294967305,4,17,layer7 cisco-jabber-
control,Input,08:55:46.917,08:55:46.917,Initiator,1,0,2,0,2046
64.103.125.3,64.103.125.29,68,4294967305,4,17,layer7
dhcp,Input,08:55:47.917,08:55:47.917,Initiator,1,0,2,0,712
10.80.101.18,10.80.101.6,5060,4294967305,4,6,layer7 cisco-collab-
control,Input,08:55:46.917,08:55:47.917,Initiator,2,23,27,12752,8773
10.1.11.4,66.102.11.99,80,4294967305,4,6,layer7 google-
services,Input,08:55:46.917,08:55:46.917,Initiator,2,3,5,1733,663
64.103.125.2,64.103.125.97,68,4294967305,4,17,layer7
dhcp,Input,08:55:47.917,08:55:53.917,Initiator,1,0,4,0,1412
64.103.125.29,64.103.101.181,67,4294967305,4,17,layer7
dhcp,Input,08:55:47.917,08:55:47.917,Initiator,1,0,1,0,350
```

Basic Troubleshooting - Questions and Answers

Following are the basic questions and answers for troubleshooting wired Application Visibility and Control:

1. **Question:** My IPv6 traffic is not being classified.

   Answer: Currently only IPv4 traffic is supported.

2. **Question:** My multicast traffic is not being classified

   Answer: Currently only unicast traffic is supported

3. **Question:** I send ping but I don’t see them being classified

   Answer: Only TCP/UDP protocols are supported

4. **Question:** Why can’t I attach NBAR to an SVI?

   Answer: NBAR is only supported on physical interfaces.

5. **Question:** I see that most of my traffic is CAPWAP traffic, why?

   Answer: Make sure that you have enabled NBAR on an access port that is not connected to a wireless access port. All traffic coming from AP’s will be classified as capwap. Actual classification in this case happens either on the AP or WLC.

6. **Question:** In protocol-discovery, I see traffic only on one side. Along with that, there are a lot of unknown traffic.

   Answer: This usually indicates that NBAR sees asymmetric traffic: one side of the traffic is classified in one switch member and the other on a different member. The recommendation is to attach NBAR only on access ports where we see both sides of the traffic. If you have multiple uplinks, you can’t attach NBAR on them due to this issue. Similar issue happens if you configure NBAR on an interface that is part of a port channel.

7. **Question:** With protocol-discovery, I see an aggregate view of all application. How can I see traffic distribution over time?

   Answer: WebUI will give you view of traffic over time for the last 48 hours.

8. **Question:** I can't configure queue-based egress policy with **match protocol** *protocol-name* command.

   Answer: Only **shape** and **set DSCP** are supported in a policy with NBAR2 based classifiers. Common practice is to set DSCP on ingress and perform shaping on egress based on DSCP.

9. **Question:** I don’t have NBAR2 attached to any interface but I still see that NBAR2 is activated.

   Answer: If you have any class-map with **match protocol** *protocol-name*, NBAR will be globally activated on the switch but no traffic will be subjected to NBAR classification. This is an expected behavior and it does not consume any resources.

10. **Question:** I see some traffic under the default QOS queue. Why?

    Answer: For each new flow, it takes a few packets to classify it and install the result in the hardware. During this time, the classification would be 'un-known' and traffic will fall under the default queue.

CHAPTER 138

SDM Template

Information About SDM Template

The switches support the Advanced template.

You can use the **show sdm prefer** privileged EXEC command which displays the SDM template in use.

Configuration Examples for SDM Templates

Example: Displaying SDM Template

The following example displays the output showing the SDM template information.

```
Device# show sdm prefer

Showing SDM Template Info

This is the Advanced template.
  Number of VLANs:                                1024
  Unicast MAC addresses:                          24576
  Overflow Unicast MAC addresses:                 256
  L2 Multicast entries:                           1024
  L3 Multicast entries:                           1024
  Overflow L3 Multicast entries:                  256
  Directly connected routes:                      8192
  Indirect routes:                                7168
  Security Access Control Entries:                1536
  QoS Access Control Entries:                     1024
  Policy Based Routing ACEs:                      640
  Netflow Input ACEs:                             128
  Netflow Output ACEs:                            128
  Flow SPAN ACEs:                                 256
  Tunnels:                                        128
  LISP Instance Mapping Entries:                  256
  Control Plane Entries:                          512
  Input Netflow flows:                            8192
  Output Netflow flows:                           8192
```

```
  SGT/DGT (or) MPLS VPN entries:                      2048
  SGT/DGT (or) MPLS VPN Overflow entries:             256
  Wired clients:                                      2048
  MACSec SPD Entries:                                 128

These numbers are typical for L2 and IPv4 features.
Some features such as IPv6, use up double the entry size;
so only half as many entries can be created.
```

CHAPTER 139

Configuring System Message Logs

Information About Configuring System Message Logs

System Messsage Logging

By default, a switch sends the output from system messages and **debug** privileged EXEC commands to a logging process. . The logging process controls the distribution of logging messages to various destinations, such as the logging buffer, terminal lines, or a UNIX syslog server, depending on your configuration. The process also sends messages to the console.

When the logging process is disabled, messages are sent only to the console. The messages are sent as they are generated, so message and debug output are interspersed with prompts or output from other commands. Messages appear on the active consoles after the process that generated them has finished.

You can set the severity level of the messages to control the type of messages displayed on the consoles and each of the destinations. You can time-stamp log messages or set the syslog source address to enhance real-time debugging and management. For information on possible messages, see the system message guide for this release.

You can access logged system messages by using the switch command-line interface (CLI) or by saving them to a properly configured syslog server. The switch software saves syslog messages in an internal buffer on a standalone switch. If a standalone switch , the log is lost unless you had saved it to flash memory.

You can remotely monitor system messages by viewing the logs on a syslog server or by accessing the switch through Telnet, through the console port, or through the Ethernet management port.

Note The syslog format is compatible with 4.3 BSD UNIX.

System Log Message Format

System log messages can contain up to 80 characters and a percent sign (%), which follows the optional sequence number or time-stamp information, if configured. Depending on the switch, messages appear in one of these formats:

- *seq no:timestamp: %facility-severity-MNEMONIC:description (hostname-n)*
- *seq no:timestamp: %facility-severity-MNEMONIC:description*

The part of the message preceding the percent sign depends on the setting of these global configuration commands:

- **service sequence-numbers**
- **service timestamps log datetime**
- **service timestamps log datetime** [**localtime**] [**msec**] [**show-timezone**]
- **service timestamps log uptime**

Table 144: System Log Message Elements

Element	Description
seq no:	Stamps log messages with a sequence number only if the **service sequence-numbers** global configuration command is configured.
timestamp formats: *mm/dd h h:mm:ss* or *hh:mm:ss* (short uptime) or *d h* (long uptime)	Date and time of the message or event. This information appears only if the **service timestamps log** [**datetime** | **log**] global configuration command is configured.
facility	The facility to which the message refers (for example, SNMP, SYS, and so forth).
severity	Single-digit code from 0 to 7 that is the severity of the message.
MNEMONIC	Text string that uniquely describes the message.
description	Text string containing detailed information about the event being reported.

Default System Message Logging Settings

Table 145: Default System Message Logging Settings

Feature	Default Setting
System message logging to the console	Enabled.

Feature	Default Setting
Console severity	Debugging.
Logging file configuration	No filename specified.
Logging buffer size	4096 bytes.
Logging history size	1 message.
Time stamps	Disabled.
Synchronous logging	Disabled.
Logging server	Disabled.
Syslog server IP address	None configured.
Server facility	Local7
Server severity	Informational.

Syslog Message Limits

If you enabled syslog message traps to be sent to an SNMP network management station by using the **snmp-server enable trap** global configuration command, you can change the level of messages sent and stored in the switch history table. You also can change the number of messages that are stored in the history table.

Messages are stored in the history table because SNMP traps are not guaranteed to reach their destination. By default, one message of the level **warning** and numerically lower levels are stored in the history table even if syslog traps are not enabled.

When the history table is full (it contains the maximum number of message entries specified with the **logging history size** global configuration command), the oldest message entry is deleted from the table to allow the new message entry to be stored.

The history table lists the level keywords and severity level. For SNMP usage, the severity level values increase by 1. For example, *emergencies* equal 1, not 0, and *critical* equals 3, not 2.

How to Configure System Message Logs

Setting the Message Display Destination Device

If message logging is enabled, you can send messages to specific locations in addition to the console.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**logging buffered** [*size*] **Example:** `Device(config)# logging buffered 8192`	Logs messages to an internal buffer on the switch. The range is 4096 to 2147483647 bytes. The default buffer size is 4096 bytes. If a standalone switchfails, the log file is lost unless you previously saved it to flash memory. See Step 4. **Note** Do not make the buffer size too large because the switch could run out of memory for other tasks. Use the **show memory** privileged EXEC command to view the free processor memory on the switch. However, this value is the maximum available, and the buffer size should *not* be set to this amount.
Step 3	**logging** *host* **Example:** `Device(config)# logging 125.1.1.100`	Logs messages to a UNIX syslog server host. *host* specifies the name or IP address of the host to be used as the syslog server. To build a list of syslog servers that receive logging messages, enter this command more than once.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**terminal monitor** **Example:** `Device# terminal monitor`	Logs messages to a nonconsole terminal during the current session. Terminal parameter-setting commands are set locally and do not remain in effect after the session has ended. You must perform this step for each session to see the debugging messages.

Synchronizing Log Messages

You can synchronize unsolicited messages and **debug** privileged EXEC command output with solicited device output and prompts for a specific console port line or virtual terminal line. You can identify the types of messages to be output asynchronously based on the level of severity. You can also configure the maximum number of buffers for storing asynchronous messages for the terminal after which messages are dropped.

When synchronous logging of unsolicited messages and **debug** command output is enabled, unsolicited device output appears on the console or printed after solicited device output appears or is printed. Unsolicited messages and **debug** command output appears on the console after the prompt for user input is returned. Therefore, unsolicited messages and **debug** command output are not interspersed with solicited device output and prompts. After the unsolicited messages appear, the console again displays the user prompt.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**line** [**console** | **vty**] *line-number* [*ending-line-number*] **Example:** `Device(config)# line console`	Specifies the line to be configured for synchronous logging of messages. • **console** —Specifies configurations that occur through the switch console port or the Ethernet management port. • **line vty** *line-number*—Specifies which vty lines are to have synchronous logging enabled. You use a vty connection for configurations that occur through a Telnet session. The range of line numbers is from 0 to 15. You can change the setting of all 16 vty lines at once by entering: line vty 0 15 You can also change the setting of the single vty line being used for your current connection. For example, to change the setting for vty line 2, enter: line vty 2 When you enter this command, the mode changes to line configuration.

	Command or Action	Purpose
Step 3	**logging synchronous** [**level** [*severity-level* | **all**] | **limit** *number-of-buffers*] **Example:** `Device(config)# logging synchronous level 3 limit 1000`	Enables synchronous logging of messages. • (Optional) **level** *severity-level*—Specifies the message severity level. Messages with a severity level equal to or higher than this value are printed asynchronously. Low numbers mean greater severity and high numbers mean lesser severity. The default is 2. • (Optional) **level all**—Specifies that all messages are printed asynchronously regardless of the severity level. • (Optional) **limit** *number-of-buffers*—Specifies the number of buffers to be queued for the terminal after which new messages are dropped. The range is 0 to 2147483647. The default is 20.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Disabling Message Logging

Message logging is enabled by default. It must be enabled to send messages to any destination other than the console. When enabled, log messages are sent to a logging process, which logs messages to designated locations asynchronously to the processes that generated the messages.

Disabling the logging process can slow down the switch because a process must wait until the messages are written to the console before continuing. When the logging process is disabled, messages appear on the console as soon as they are produced, often appearing in the middle of command output.

The **logging synchronous** global configuration command also affects the display of messages to the console. When this command is enabled, messages appear only after you press **Return**.

To reenable message logging after it has been disabled, use the **logging on** global configuration command.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 2	**no logging console** **Example:** `Device(config)# no logging console`	Disables message logging.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Enabling and Disabling Time Stamps on Log Messages

By default, log messages are not time-stamped.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	Use one of these commands: • **service timestamps log uptime** • **service timestamps log datetime**[**msec** | **localtime** | **show-timezone**] **Example:** `Device(config)# service timestamps log uptime` or `Device(config)# service timestamps log datetime`	Enables log time stamps. • **log uptime**—Enables time stamps on log messages, showing the time since the system was rebooted. • **log datetime**—Enables time stamps on log messages. Depending on the options selected, the time stamp can include the date, time in milliseconds relative to the local time zone, and the time zone name.
Step 3	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	

Enabling and Disabling Sequence Numbers in Log Messages

If there is more than one log message with the same time stamp, you can display messages with sequence numbers to view these messages. By default, sequence numbers in log messages are not displayed.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**service sequence-numbers** **Example:** `Device(config)# service sequence-numbers`	Enables sequence numbers.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Defining the Message Severity Level

Limit messages displayed to the selected device by specifying the severity level of the message.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 2	**logging console** *level* **Example:** `Device(config)# logging console 3`	Limits messages logged to the console. By default, the console receives debugging messages and numerically lower levels.
Step 3	**logging monitor** *level* **Example:** `Device(config)# logging monitor 3`	Limits messages logged to the terminal lines. By default, the terminal receives debugging messages and numerically lower levels.
Step 4	**logging trap** *level* **Example:** `Device(config)# logging trap 3`	Limits messages logged to the syslog servers. By default, syslog servers receive informational messages and numerically lower levels.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Limiting Syslog Messages Sent to the History Table and to SNMP

This task explains how to limit syslog messages that are sent to the history table and to SNMP.

This task is optional.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**logging history** *level* **Example:** `Device(config)# logging history 3`	Changes the default level of syslog messages stored in the history file and sent to the SNMP server. By default, **warnings**, **errors**, **critical**, **alerts**, and **emergencies** messages are sent.

	Command or Action	Purpose
Step 3	**logging history size** *number* **Example:** `Device(config)# logging history size 200`	Specifies the number of syslog messages that can be stored in the history table. The default is to store one message. The range is 0 to 500 messages.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Logging Messages to a UNIX Syslog Daemon

This task is optional.

Note Some recent versions of UNIX syslog daemons no longer accept by default syslog packets from the network. If this is the case with your system, use the UNIX **man syslogd** command to decide what options must be added to or removed from the syslog command line to enable logging of remote syslog messages.

Before you begin

- Log in as root.
- Before you can send system log messages to a UNIX syslog server, you must configure the syslog daemon on a UNIX server.

Procedure

	Command or Action	Purpose
Step 1	Add a line to the file /etc/syslog.conf. **Example:** **local7.debug /usr/adm/logs/***cisco.log*	• **local7**—Specifies the logging facility. • **debug**—Specifies the syslog level. The file must already exist, and the syslog daemon must have permission to write to it.
Step 2	Enter these commands at the UNIX shell prompt. **Example:** `$ touch /var/log/cisco.log` `$ chmod 666 /var/log/cisco.log`	Creates the log file. The syslog daemon sends messages at this level or at a more severe level to this file.

	Command or Action	Purpose
Step 3	Make sure the syslog daemon reads the new changes. **Example:** `$ kill -HUP `cat /etc/syslog.pid``	For more information, see the **man syslog.conf** and **man syslogd** commands on your UNIX system.

Monitoring and Maintaining System Message Logs

Monitoring Configuration Archive Logs

Command	Purpose
show archive log config {**all** | **number** [*end-number*] | **user** *username* [**session** *number*] *number* [*end-number*] | **statistics**} [**provisioning**]	Displays the entire configuration log or the log for specified parameters.

Configuration Examples for System Message Logs

Example: Switch System Message

This example shows a partial switch system message on a switch:

```
00:00:46: %LINK-3-UPDOWN: Interface Port-channel1, changed state to up
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet1/1, changed state to up
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet1/2, changed state to up
00:00:48: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state to down
00:00:48: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet1/1, changed state
 to down 2
*Mar  1 18:46:11: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
18:47:02: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Mar  1 18:48:50.483 UTC: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
```

CHAPTER 140

Configuring Online Diagnostics

Restrictions for Online Diagnostics

MACsec diagnostic test is not supported on half duplex mode (interfaces operating at 10 or 100 Mb/s).

Information About Configuring Online Diagnostics

With online diagnostics, you can test and verify the hardware functionality of a device while the device is connected to a live network. Online diagnostics contains packet-switching tests that check different hardware components and verify the data path and control signals.

Online diagnostics detects problems in these areas:

- Hardware components
- Interfaces (Ethernet ports and so forth)
- Solder joints

Online diagnostics are categorized as on-demand, scheduled, or health-monitoring diagnostics. On-demand diagnostics run from the CLI; scheduled diagnostics run at user-designated intervals or at specified times when the device is connected to a live network; and health-monitoring runs in the background with user-defined intervals. The health-monitoring test runs every 90, 100, or 150 seconds based on the test.

After you configure online diagnostics, you can manually start diagnostic tests or display the test results. You can also see which tests are configured for the device and the diagnostic tests that have already run.

Generic Online Diagnostics (GOLD) Tests

Note

- Before you enable online diagnostics tests, enable console logging to see all the warning messages.
- While tests are running, all the ports are shut down because a stress test is being performed with looping ports internally, and external traffic might affect the test results. Reboot the switch to bring it to normal operation. When you run the command to reload a switch, the system will ask you if the configuration should be saved. Do not save the configuration.
- If you are running tests on other modules, after a test is initiated and complete, you must reset the module.

The following sections provide information about GOLD tests.

DiagGoldPktTest

This GOLD packet loopback test verifies the MAC-level loopback functionality. In this test, a GOLD packet is sent, for which Unified Access Data Plane (UADP) ASIC provides support in the hardware. The packet loops back at MAC-level and is matched against the stored packet.

Attribute	Description
Disruptive or Nondisruptive	Nondisruptive.
Recommendation	Run this on-demand test as per requirement.
Default	Off.
Corrective action	–
Hardware support	All modules.

DiagThermalTest

This test verifies the temperature reading from a device sensor.

Attribute	Description
Disruptive or Nondisruptive	Nondisruptive.
Recommendation	Do not disable. Run this as an on-demand test, and as a health-monitoring test if the administrator is down.
Default	On.
Corrective action	–
Hardware support	All modules.

DiagPhyLoopbackTest

This PHY loopback test verifies the PHY-level loopback functionality. In this test, a packet, which loops back at the PHY level and is matched against the stored packet, is sent. It cannot be run as a health-monitoring test.

Note In certain cases when this test is run on-demand, ports are moved to the error-disabled state. In such cases, use the **shut** and **no shut** command in interface configuration mode to reenable these ports.

Attribute	Description
Disruptive or Nondisruptive	Disruptive.
Recommendation	If the link to the external connector is down, run this on-demand test to check the health of the link.
Default	Off.
Corrective action	–
Hardware support	All modules.

DiagScratchRegisterTest

This Scratch Register test monitors the health of ASICs by writing values into registers, and reading back the values from these registers.

Attribute	Description
Disruptive or Nondisruptive	Nondisruptive.
Recommendation	Do not disable. Run this test if the task of writing values to the registers fails. This can be run as a health-monitoring test and also as an on-demand test.
Default	On.
Corrective action	–
Hardware support	All modules.

TestUnusedPortLoopback

This test verifies the PHY-level loopback functionality for admin-down ports. In this test, a packet which loops back at the PHY level and is matched against the stored packet, is sent.

Attribute	Description
Disruptive or Nondisruptive	Nondisruptive.
Recommendation	This can be run as a health-monitoring test and also as an on-demand test.
Default	Off.
Corrective action	Displays a syslog message if the test fails for a port.
Hardware support	All modules.

How to Configure Online Diagnostics

The following sections provide information about the various procedures that comprise the online diagnostics configuration.

Starting Online Diagnostic Tests

After you configure diagnostic tests to run on a device, use the **diagnostic start** privileged EXEC command to begin diagnostic testing.

After starting the tests, you cannot stop the testing process midway.

Use the **diagnostic start switch** privileged EXEC command to manually start online diagnostic testing:

Procedure

	Command or Action	Purpose
Step 1	**diagnostic start switch** *number* **test** {*name* | *test-id* | *test-id-range* | **all** | **basic** | **complete** | **minimal** | **non-disruptive** | **per-port**} **Example:** `Device# diagnostic start switch 2 test basic`	Starts the diagnostic tests. You can specify the tests by using one of these options: • *name*: Enters the name of the test. • *test-id*: Enters the ID number of the test. • *test-id-range*: Enters the range of test IDs by using integers separated by a comma and a hyphen. • **all**: Starts all of the tests. • **basic**: Starts the basic test suite. • **complete**: Starts the complete test suite. • **minimal**: Starts the minimal bootup test suite. • **non-disruptive**: Starts the nondisruptive test suite. • **per-port**: Starts the per-port test suite.

Configuring Online Diagnostics

You must configure the failure threshold and the interval between tests before enabling diagnostic monitoring.

Scheduling Online Diagnostics

You can schedule online diagnostics to run at a designated time of day, or on a daily, weekly, or monthly basis for a device. Use the **no** form of the **diagnostic schedule switch** command to remove the scheduling.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device #configure terminal`	Enters global configuration mode.
Step 2	**diagnostic schedule** *number* **test** {*name* | *test-id* | *test-id-range* | **all** | **basic** | **complete** | **minimal** | **non-disruptive** | **per-port**} {**daily** | **on** *mm dd yyyy hh:mm* | **port** *inter-port-number port-number-list* | **weekly** *day-of-week hh:mm*} **Example:** `Device(config)# diagnostic schedule 3 test 1-5 on July 3 2013 23:10`	Schedules on-demand diagnostic test for a specific day and time. When specifying the test to be scheduled, use these options: • *name*: Name of the test that appears in the **show diagnostic content** command output. • *test-id*: ID number of the test that appears in the **show diagnostic content** command output. • *test-id-range*: ID numbers of the tests that appear in the **show diagnostic content** command output. • **all**: All test IDs. • **basic**: Starts the basic on-demand diagnostic tests. • **complete**: Starts the complete test suite. • **minimal**: Starts the minimal bootup test suite. • **non-disruptive**: Starts the nondisruptive test suite. • **per-port**: Starts the per-port test suite. You can schedule the tests as follows: • Daily: Use the **daily** *hh:mm* parameter. • Specific day and time: Use the **on** *mm dd yyyy hh:mm* parameter. • Weekly: Use the **weekly** *day-of-week hh:mm* parameter.

Configuring Health-Monitoring Diagnostics

You can configure health-monitoring diagnostic testing on a device while it is connected to a live network. You can configure the execution interval for each health-monitoring test, enable the device to generate a syslog message because of a test failure, and enable a specific test.

Use the **no** form of this command to disable testing.

By default, health monitoring is enabled only for a few tests, and the device generates a syslog message when a test fails.

Follow these steps to configure and enable the health-monitoring diagnostic tests:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**diagnostic monitor interval switch** *number* **test** {*name* | *test-id* | *test-id-range* | **all**} *hh:mm:ss milliseconds day* **Example:** `Device(config)# diagnostic monitor interval switch 2 test 1 12:30:00 750 5`	Configures the health-monitoring interval of the specified test. When specifying a test, use one of these parameters: • *name*: Name of the test that appears in the **show diagnostic content** command output. • *test-id*: ID number of the test that appears in the **show diagnostic content** command output. • *test-id-range*: ID numbers of the tests that appear in the **show diagnostic content** command output. • **all**: All the diagnostic tests. When specifying the interval, set these parameters: • *hh:mm:ss*: Monitoring interval, in hours, minutes, and seconds. The range for *hh*

	Command or Action	Purpose
		is 0 to 24, and the range for *mm* and *ss* is 0 to 60. • *milliseconds*: Monitoring interval, in milliseconds (ms). The range is from 0 to 999. • *day*: Monitoring interval, in number of days. The range is from 0 to 20.
Step 4	**diagnostic monitor syslog** **Example:** `Device(config)# diagnostic monitor syslog`	(Optional) Configures the switch to generate a syslog message when a health-monitoring test fails.
Step 5	**diagnostic monitor threshold switch** *number number* **test** {*name* | *test-id* | *test-id-range* | **all**} **failure count** *count* **Example:** `Device(config)# diagnostic monitor threshold switch 2 test 1 failure count 20`	(Optional) Sets the failure threshold for the health-monitoring test. When specifying the tests, use one of these parameters: • *name*: Name of the test that appears in the **show diagnostic content** command output. • *test-id*: ID number of the test that appears in the **show diagnostic content** command output. • *test-id-range*: ID numbers of the tests that appear in the **show diagnostic content** command output. • **all**: All the diagnostic tests. The range for the failure threshold *count* is 0 to 99.
Step 6	**diagnostic monitor switch***number* **test** {*name* | *test-id* | *test-id-range* | **all**} **Example:** `Device(config)# diagnostic monitor switch 2 test 1`	Enables the specified health-monitoring tests. The **switch** *number* keyword is supported only on stacking switches. When specifying the tests, use one of these parameters: • *name*: Name of the test that appears in the **show diagnostic content** command output. • *test-id*: ID number of the test that appears in the **show diagnostic content** command output.

	Command or Action	Purpose
		• *test-id-range*: ID numbers of the tests that appear in the **show diagnostic content** command output. • **all**: All the diagnostic tests.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show diagnostic** { **content** | **post** | **result** | **schedule** | **status** | **switch** }	(Optional) Display the online diagnostic test results and the supported test suites.
Step 9	**show running-config** **Example:** `Device# show running-config`	(Optional) Verifies your entries.
Step 10	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring and Maintaining Online Diagnostics

You can display the online diagnostic tests that are configured for a device and check the test results by using the privileged EXEC **show** commands in this table:

Table 146: Commands for Diagnostic Test Configuration and Results

Command	Purpose
show diagnostic content switch [*number* | **all**]	Displays the online diagnostics configured for a switch.
show diagnostic status	Displays the diagnostic tests that are running currently. .
show diagnostic result switch [*number* | **all**] [**detail** | **test** {*name* | *test-id* | *test-id-range* | **all**} [**detail**]]	Displays the online diagnostics test results.
show diagnostic switch [*number* | **all**] [**detail**]	Displays the online diagnostics test results.
show diagnostic schedule [*number* | **all**]	Displays the online diagnostics test schedule.

Command	Purpose
show diagnostic post	Displays the POST results. (The output is the same as the **show post** command output.)
show diagnostic events {**event-type** | **module**}	Displays diagnostic events such as error, information, or warning based on the test result.
show diagnostic description module [**number**] **test** { **name** | **test-id** | **all** }	Displays the short description of the results from an individual test or all the tests.

Configuration Examples for Online Diagnostics

The following sections provide examples of online diagnostics configurations.

Example: Configure a Health-Monitoring Test

This example shows how to configure a health-monitoring test:

```
Device(config)# diagnostic monitor threshold switch 1 test 1 failure count 50
Device(config)# diagnostic monitor interval switch 1 test TestPortAsicStackPortLoopback
```

Example: Schedule Diagnostic Test

This example shows how to schedule diagnostic testing for a specific day and time on a specific switch:

```
Device(config)# diagnostic schedule test DiagThermalTest on June 3 2013  22:25
```

This example shows how to schedule diagnostic testing to occur weekly at a certain time on a specific switch:

```
Device(config)# diagnostic schedule switch 1 test 1,2,4-6 weekly saturday 10:30
```

Example: Displaying Online Diagnostics

This example shows how to display on-demand diagnostic settings:

```
Device# show diagnostic ondemand settings

Test iterations = 1
Action on test failure = continue
```

This example shows how to display diagnostic events for errors:

```
Device# show diagnostic events event-type error

Diagnostic events (storage for 500 events, 0 events recorded)
Number of events matching above criteria = 0

No diagnostic log entry exists.
```

This example shows how to display the description for a diagnostic test:

```
Device# show diagnostic description switch 1 test all

DiagGoldPktTest :
        The GOLD packet Loopback test verifies the MAC level loopback
        functionality. In this test, a GOLD packet, for which doppler
        provides the support in hardware, is sent. The packet loops back
        at MAC level and is matched against the stored packet. It is a non
        -disruptive test.

DiagThermalTest :
        This test verifies the temperature reading from the sensor is below the yellow
        temperature threshold. It is a non-disruptive test and can be run as a health
monitoring test.

DiagPhyLoopbackTest :
        The PHY Loopback test verifies the PHY level loopback
        functionality. In this test, a packet is sent which loops back
        at PHY level and is matched against the stored packet. It is a
        disruptive test and cannot be run as a health monitoring test.

DiagScratchRegisterTest :
        The Scratch Register test monitors the health of application-specific
        integrated circuits (ASICs) by writing values into registers and reading
        back the values from these registers. It is a non-disruptive test and can
        be run as a health monitoring test.

DiagPoETest :
        This test checks the PoE controller functionality. This is a disruptive test
        and should not be performed during normal switch operation.



Device#
```

CHAPTER 141

Managing Configuration Files

Prerequisites for Managing Configuration Files

- You should have at least a basic familiarity with the Cisco IOS environment and the command-line interface.
- You should have at least a minimal configuration running on your system. You can create a basic configuration file using the **setup** command.

Restrictions for Managing Configuration Files

- Many of the Cisco IOS commands described in this document are available and function only in certain configuration modes on the device.
- Some of the Cisco IOS configuration commands are only available on certain device platforms, and the command syntax may vary on different platforms.

Information About Managing Configuration Files

Types of Configuration Files

Configuration files contain the Cisco IOS software commands used to customize the functionality of your Cisco device. Commands are parsed (translated and executed) by the Cisco IOS software when the system is booted (from the startup-config file) or when you enter commands at the CLI in a configuration mode.

Startup configuration files (startup-config) are used during system startup to configure the software. Running configuration files (running-config) contain the current configuration of the software. The two configuration files can be different. For example, you may want to change the configuration for a short time period rather

than permanently. In this case, you would change the running configuration using the **configure terminal** EXEC command but not save the configuration using the **copy running-config startup-config** EXEC command.

To change the running configuration, use the **configure terminal** command, as described in the Modifying the Configuration File, on page 2115 section. As you use the Cisco IOS configuration modes, commands generally are executed immediately and are saved to the running configuration file either immediately after you enter them or when you exit a configuration mode.

To change the startup configuration file, you can either save the running configuration file to the startup configuration using the **copy running-config startup-config** EXEC command or copy a configuration file from a file server to the startup configuration (see the "Copying a Configuration File from a TFTP Server to the Router" section for more information).

Configuration Mode and Selecting a Configuration Source

To enter configuration mode on the device, enter the **configure** command at the privileged EXEC prompt. The Cisco IOS software responds with the following prompt asking you to specify the terminal, memory, or a file stored on a network server (network) as the source of configuration commands:

```
Configuring from terminal, memory, or network [terminal]?
```

Configuring from the terminal allows you to enter configuration commands at the command line, as described in the following section. See the "Re-executing the Configuration Commands in the Startup Configuration File" section for more information.

Configuring from the network allows you to load and execute configuration commands over the network. See the "Copying a Configuration File from a TFTP Server to the Switch" section for more information.

Configuration File Changes Using the CLI

The Cisco IOS software accepts one configuration command per line. You can enter as many configuration commands as you want. You can add comments to a configuration file describing the commands you have entered. Precede a comment with an exclamation point (!). Because comments are *not* stored in NVRAM or in the active copy of the configuration file, comments do not appear when you list the active configuration with the **show running-config** or **more system:running-config** EXEC command. Comments are not displayed when you list the startup configuration with the **show startup-config** or **more nvram:startup-config** EXEC mode command. Comments are stripped out of the configuration file when it is loaded onto the device. However, you can list the comments in configuration files stored on a File Transfer Protocol (FTP), Remote Copy Protocol (RCP), or Trivial File Transfer Protocol (TFTP) server. When you configure the software using the CLI, the software executes the commands as you enter them.

Location of Configuration Files

Configuration files are stored in the following locations:

- The running configuration is stored in RAM.
- On all platforms except the Class A Flash file system platforms, the startup configuration is stored in nonvolatile random-access memory (NVRAM).

- On Class A Flash file system platforms, the startup configuration is stored in the location specified by the CONFIG_FILE environment variable (see the Specifying the CONFIG_FILE Environment Variable on Class A Flash File Systems , on page 2135 section). The CONFIG_FILE variable defaults to NVRAM and can be a file in the following file systems:
  - **nvram:** (NVRAM)
  - **flash:** (internal flash memory)
  - **usbflash0:** (external usbflash file system)
  - **usbflash1:** (external usbflash file system)

Copy Configuration Files from a Network Server to the Device

You can copy configuration files from a TFTP, rcp, or FTP server to the running configuration or startup configuration of the device. You may want to perform this function for one of the following reasons:

- To restore a backed-up configuration file.
- To use the configuration file for another device. For example, you may add another device to your network and want it to have a similar configuration to the original device. By copying the file to the new device, you can change the relevant parts rather than recreating the whole file.
- To load the same configuration commands on to all of the devices in your network so that all of the devices have similar configurations.

The **copy**{**ftp:** | **rcp:** | **tftp:system:running-config**} EXEC command loads the configuration files into the device as if you were typing the commands on the command line. The device does not erase the existing running configuration before adding the commands. If a command in the copied configuration file replaces a command in the existing configuration file, the existing command is erased. For example, if the copied configuration file contains a different IP address in a particular command than the existing configuration, the IP address in the copied configuration is used. However, some commands in the existing configuration may not be replaced or negated. In this case, the resulting configuration file is a mixture of the existing configuration file and the copied configuration file, with the copied configuration file having precedence.

To restore a configuration file to an exact copy of a file stored on a server, you need to copy the configuration file directly to the startup configuration (using the **copy ftp:**| **rcp:**| **tftp:**} **nvram:startup-config** command) and reload the device.

To copy configuration files from a server to a device, perform the tasks described in the following sections.

The protocol that you use depends on which type of server you are using. The FTP and rcp transport mechanisms provide faster performance and more reliable delivery of data than TFTP. These improvements are possible because the FTP and rcp transport mechanisms are built on and use the TCP/IP stack, which is connection-oriented.

Copying a Configuration File from the Device to a TFTP Server

In some implementations of TFTP, you must create a dummy file on the TFTP server and give it read, write, and execute permissions before copying a file over it. Refer to your TFTP documentation for more information.

Copying a Configuration File from the Device to an RCP Server

You can copy a configuration file from the device to an RCP server.

One of the first attempts to use the network as a resource in the UNIX community resulted in the design and implementation of the remote shell protocol, which included the remote shell (rsh) and remote copy (rcp) functions. Rsh and rcp give users the ability to execute commands remotely and copy files to and from a file system residing on a remote host or server on the network. The Cisco implementation of rsh and rcp interoperates with standard implementations.

The rcp **copy** commands rely on the rsh server (or daemon) on the remote system. To copy files using rcp, you need not create a server for file distribution, as you do with TFTP. You need only to have access to a server that supports the remote shell (rsh). (Most UNIX systems support rsh.) Because you are copying a file from one place to another, you must have read permission on the source file and write permission on the destination file. If the destination file does not exist, rcp creates it for you.

Although the Cisco rcp implementation emulates the functions of the UNIX rcp implementation—copying files among systems on the network—the Cisco command syntax differs from the UNIX rcp command syntax. The Cisco rcp support offers a set of **copy** commands that use rcp as the transport mechanism. These rcp **copy** commands are similar in style to the Cisco TFTP **copy** commands, but they offer an alternative that provides faster performance and reliable delivery of data. These improvements are possible because the rcp transport mechanism is built on and uses the TCP/IP stack, which is connection-oriented. You can use rcp commands to copy system images and configuration files from the device to a network server and vice versa.

You also can enable rcp support to allow users on remote systems to copy files to and from the device.

To configure the Cisco IOS software to allow remote users to copy files to and from the device, use the **ip rcmd rcp-enable** global configuration command.

Restrictions

The RCP protocol requires a client to send a remote username on each RCP request to a server. When you copy a configuration file from the device to a server using RCP, the Cisco IOS software sends the first valid username it encounters in the following sequence:

1. The username specified in the **copy** EXEC command, if a username is specified.
2. The username set by the **ip rcmd remote-username** global configuration command, if the command is configured.
3. The remote username associated with the current tty (terminal) process. For example, if the user is connected to the device through Telnet and was authenticated through the **username** command, the device software sends the Telnet username as the remote username.
4. The device host name.

For the RCP copy request to execute successfully, an account must be defined on the network server for the remote username. If the server has a directory structure, the configuration file or image is written to or copied from the directory associated with the remote username on the server. For example, if the system image resides in the home directory of a user on the server, you can specify that user name as the remote username.

Use the **ip rcmd remote-username** command to specify a username for all copies. (Rcmd is a UNIX routine used at the super-user level to execute commands on a remote machine using an authentication scheme based on reserved port numbers. Rcmd stands for “remote command”). Include the username in the **copy** command if you want to specify a username for that copy operation only.

If you are writing to the server, the RCP server must be properly configured to accept the RCP write request from the user on the device. For UNIX systems, you must add an entry to the .rhosts file for the remote user on the RCP server. For example, suppose the device contains the following configuration lines:

```
hostname Device1
ip rcmd remote-username User0
```

If the device IP address translates to device1.example.com, then the .rhosts file for User0 on the RCP server should contain the following line:

```
Device1.example.com Device1
```

Requirements for the RCP Username

The RCP protocol requires a client to send a remote username on each RCP request to a server. When you copy a configuration file from the device to a server using RCP, the Cisco IOS software sends the first valid username it encounters in the following sequence:

1. The username specified in the **copy** EXEC command, if a username is specified.
2. The username set by the **ip rcmd remote-username** global configuration command, if the command is configured.
3. The remote username associated with the current tty (terminal) process. For example, if the user is connected to the device through Telnet and is authenticated through the **username** command, the device software sends the Telnet username as the remote username.
4. The device host name.

For the RCP copy request to execute, an account must be defined on the network server for the remote username. If the server has a directory structure, the configuration file or image is written to or copied from the directory associated with the remote username on the server. For example, if the system image resides in the home directory of a user on the server, specify that user name as the remote username.

Refer to the documentation for your RCP server for more information.

Copying a Configuration File from the Device to an FTP Server

You can copy a configuration file from the device to an FTP server.

Understanding the FTP Username and Password

Note The password must not contain the special character '@'. If the character '@' is used, the copy fails to parse the IP address of the server.

The FTP protocol requires a client to send a remote username and password on each FTP request to a server. When you copy a configuration file from the device to a server using FTP, the Cisco IOS software sends the first valid username it encounters in the following sequence:

1. The username specified in the **copy** EXEC command, if a username is specified.
2. The username set by the **ip ftp username** global configuration command, if the command is configured.

3. Anonymous.

The device sends the first valid password it encounters in the following sequence:

1. The password specified in the **copy** command, if a password is specified.
2. The password set by the **ip ftp password** command, if the command is configured.
3. The device forms a password *username @devicename.domain* . The variable *username* is the username associated with the current session, *devicename* is the configured host name, and *domain* is the domain of the device.

The username and password must be associated with an account on the FTP server. If you are writing to the server, the FTP server must be properly configured to accept the FTP write request from the user on the device.

If the server has a directory structure, the configuration file or image is written to or copied from the directory associated with the username on the server. For example, if the system image resides in the home directory of a user on the server, specify that user name as the remote username.

Refer to the documentation for your FTP server for more information.

Use the **ip ftp username** and **ip ftp password** global configuration commands to specify a username and password for all copies. Include the username in the **copy** EXEC command if you want to specify a username for that copy operation only.

Copying files through a VRF

You can copy files through a VRF interface specified in the **copy** command. Specifying the VRF in the **copy** command is easier and more efficient as you can directly change the source interface without using a change request for the configuration.

Example

The following example shows how to copy files through a VRF, using the **copy** command:

```
Device#
Address or name of remote host [10.1.2.3]?
Source username [ScpUser]?
Source filename [/auto/tftp-server/ScpUser/vrf_test.txt]?
Destination filename [vrf_test.txt]?
Getting the vrf name as test-vrf
Password:
Sending file modes: C0644 10 vrf_test.txt
!
223 bytes copied in 22.740 secs (10 bytes/sec)
```

Copy Configuration Files from a Switch to Another Switch

You can copy the configurations from one switch to another. This is a 2-step process - Copy the configurations from the switch to the TFTP server, and then from TFTP to another switch.

To copy your current configurations from the switch, run the command **copy startup-config tftp:** and follow the instructions. The configurations are copied onto the TFTP server.

Then, login to another switch and run the command **copy tftp: startup-config** and follow the instructions. The configurations are now copied onto the other switch.

After the configurations are copied, to save your configurations, use **write memory** command and then either reload the switch or run the **copy startup-config running-config** command

Configuration Files Larger than NVRAM

To maintain a configuration file that exceeds the size of NVRAM, you should be aware of the information in the following sections.

Compressing the Configuration File

The **service compress-config** global configuration command specifies that the configuration file be stored compressed in NVRAM. Once the configuration file has been compressed, the device functions normally. When the system is booted, it recognizes that the configuration file is compressed, expands it, and proceeds normally. The **more nvram:startup-config** EXEC command expands the configuration before displaying it.

Before you compress configuration files, refer to the appropriate hardware installation and maintenance publication. Verify that your system's ROMs support file compression. If not, you can install new ROMs that support file compression.

The size of the configuration must not exceed three times the NVRAM size. For a 128-KB size NVRAM, the largest expanded configuration file size is 384 KB.

Storing the Configuration in Flash Memory on Class A Flash File Systems

On class A Flash file system devices, you can store the startup configuration in flash memory by setting the **CONFIG_FILE** environment variable to a file in internal flash memory or flash memory in a PCMCIA slot.

See the Specifying the CONFIG_FILE Environment Variable on Class A Flash File Systems , on page 2135 section for more information.

Care must be taken when editing or changing a large configuration. Flash memory space is used every time a **copy system:running-config nvram:startup-config** EXEC command is issued. Because file management for flash memory (such as optimizing free space) is not done automatically, you must pay close attention to available flash memory. Use the **squeeze** command to reclaim used space. We recommend that you use a large-capacity Flash card of at least 20 MB.

Loading the Configuration Commands from the Network

You can also store large configurations on FTP, RCP, or TFTP servers and download them at system startup. To use a network server to store large configurations, see the Copying a Configuration File from the Device to a TFTP Server , on page 2116 and Configuring the Device to Download Configuration Files, on page 2113 sections for more information on these commands.

Configuring the Device to Download Configuration Files

You can configure the device to load one or two configuration files at system startup. The configuration files are loaded into memory and read in as if you were typing the commands at the command line. Thus, the configuration for the device is a mixture of the original startup configuration and the one or two downloaded configuration files.

Network Versus Host Configuration Files

For historical reasons, the first file the device downloads is called the network configuration file. The second file the device downloads is called the host configuration file. Two configuration files can be used when all of the devices on a network use many of the same commands. The network configuration file contains the

standard commands used to configure all of the devices. The host configuration files contain the commands specific to one particular host. If you are loading two configuration files, the host configuration file should be the configuration file you want to have precedence over the other file. Both the network and host configuration files must reside on a network server reachable via TFTP, RCP, or FTP, and must be readable.

How to Manage Configuration File Information

Displaying Configuration File Information

To display information about configuration files, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show boot** **Example:** `Device# show boot`	Lists the contents of the BOOT environment variable (if set), the name of the configuration file pointed to by the CONFIG_FILE environment variable, and the contents of the BOOTLDR environment variable.
Step 3	**more** *file-url* **Example:** `Device# more 10.1.1.1`	Displays the contents of a specified file.
Step 4	**show running-config** **Example:** `Device# show running-config`	Displays the contents of the running configuration file. (Command alias for the **more system:running-config** command.)
Step 5	**show startup-config** **Example:** `Device# show startup-config`	Displays the contents of the startup configuration file. (Command alias for the **more nvram:startup-config** command.) On all platforms except the Class A Flash file system platforms, the default startup-config file usually is stored in NVRAM. On the Class A Flash file system platforms, the CONFIG_FILE environment variable points to the default startup-config file. The CONFIG_FILE variable defaults to NVRAM.

Modifying the Configuration File

The Cisco IOS software accepts one configuration command per line. You can enter as many configuration commands as you want. You can add comments to a configuration file describing the commands you have entered. Precede a comment with an exclamation point (!). Because comments are *not* stored in NVRAM or in the active copy of the configuration file, comments do not appear when you list the active configuration with the **show running-config** or **more system:running-config** EXEC commands. Comments do not display when you list the startup configuration with the **show startup-config** or **more nvram:startup-config** EXEC mode commands. Comments are stripped out of the configuration file when it is loaded onto the device. However, you can list the comments in configuration files stored on a File Transfer Protocol (FTP), Remote Copy Protocol (RCP), or Trivial File Transfer Protocol (TFTP) server. When you configure the software using the CLI, the software executes the commands as you enter them. To configure the software using the CLI, use the following commands in privileged EXEC mode:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	*configuration-command* **Example:** `Device(config)# <configuration command>`	Enter any configuration command. The Cisco IOS documentation set describes configuration commands organized by technology.
Step 4	**end** **Example:** `Device(config)# end`	Ends the configuration session and exits to EXEC mode.
Step 5	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the running configuration file as the startup configuration file. You may also use the **copy running-config startup-config** command alias, but you should be aware that this command is less precise. On most platforms, this command saves the configuration to NVRAM. On the Class A Flash file system platforms, this step saves the configuration to the location specified by the CONFIG_FILE environment variable (the

	Command or Action	Purpose
		default CONFIG_FILE variable specifies that the file should be saved to NVRAM).

Examples

In the following example, the device prompt name of the device is configured. The comment line, indicated by the exclamation mark (!), does not execute any command. The **hostname** command is used to change the device name from device to new_name. By pressing Ctrl-Z (^Z) or entering the **end** command, the user quits configuration mode. The **copy system:running-config nvram:startup-config** command saves the current configuration to the startup configuration.

```
Device# configure terminal
Device(config)# !The following command provides the switch host name.
Device(config)# hostname new_name
new_name(config)# end
new_name# copy system:running-config nvram:startup-config
```

When the startup configuration is NVRAM, it stores the current configuration information in text format as configuration commands, recording only non-default settings. The memory is checksummed to guard against corrupted data.

Note Some specific commands might not get saved to NVRAM. You need to enter these commands again if you reboot the machine. These commands are noted in the documentation. We recommend that you keep a list of these settings so that you can quickly reconfigure your device after rebooting.

Copying a Configuration File from the Device to a TFTP Server

To copy configuration information on a TFTP network server, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy system:running-config tftp:** [[[*//location*]*/directory*]*/filename*] **Example:** `Device# copy system:running-config tftp: //server1/topdir/file10`	Copies the running configuration file to a TFTP server.

	Command or Action	Purpose
Step 3	**copy nvram:startup-config tftp:** [[[*//location*]*/directory*]*/filename*] **Example:** `Device# copy nvram:startup-config tftp: //server1/1stdir/file10`	Copies the startup configuration file to a TFTP server.

Examples

The following example copies a configuration file from a device to a TFTP server:

```
Device# copy system:running-config tftp://172.16.2.155/tokyo-confg
Write file tokyo-confg on host 172.16.2.155? [confirm] Y
Writing tokyo-confg!!! [OK]
```

What to Do Next

After you have issued the **copy** command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from the Device to an RCP Server

To copy a startup configuration file or a running configuration file from the device to an RCP server, use the following commands beginning in privileged EXEC mode:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username NetAdmin1`	(Optional) Changes the default remote username.

	Command or Action	Purpose
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode.
Step 5	Do one of the following: • **copy system:running-config rcp:** [[[//[*username@*]*location*]/*directory*]/*filename*] • **copy nvram:startup-config rcp:** [[[//[*username@*]*location*]/*directory*]/*filename*] **Example:** `Device# copy system:running-config rcp: //NetAdmin1@example.com/dir-files/file1`	• Specifies that the device running configuration file is to be stored on an RCP server or • Specifies that the device startup configuration file is to be stored on an RCP server

Examples

Storing a Running Configuration File on an RCP Server

The following example copies the running configuration file named runfile2-confg to the netadmin1 directory on the remote host with an IP address of 172.16.101.101:

```
Device# copy system:running-config rcp://netadmin1@172.16.101.101/runfile2-confg
Write file runfile2-confg on host 172.16.101.101?[confirm]
Building configuration...[OK]
Connected to 172.16.101.101
Device#
```

Storing a Startup Configuration File on an RCP Server

The following example shows how to store a startup configuration file on a server by using RCP to copy the file:

```
Device# configure terminal

Device(config)# ip rcmd remote-username netadmin2

Device(config)# end

Device# copy nvram:startup-config rcp:

Remote host[]? 172.16.101.101

Name of configuration file to write [start-confg]?
Write file start-confg on host 172.16.101.101?[confirm]
![OK]
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from the Device to the FTP Server

To copy a startup configuration file or a running configuration file from the device to an FTP server, complete the following tasks:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode on the device.
Step 3	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username NetAdmin1`	(Optional) Specifies the default remote username.
Step 4	**ip ftp password** *password* **Example:** `Device(config)# ip ftp password adminpassword`	(Optional) Specifies the default password.
Step 5	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode. This step is required only if you override the default remote username or password (see Steps 2 and 3).
Step 6	Do one of the following: • **copy system:running-config ftp:** [[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] or • **copy nvram:startup-config ftp:** [[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] **Example:**	Copies the running configuration or startup configuration file to the specified location on the FTP server.

	Command or Action	Purpose
	`Device# copy system:running-config ftp:`	

Examples

Storing a Running Configuration File on an FTP Server

The following example copies the running configuration file named runfile-confg to the netadmin1 directory on the remote host with an IP address of 172.16.101.101:

```
Device# copy system:running-config ftp://netadmin1:mypass@172.16.101.101/runfile-confg
Write file runfile-confg on host 172.16.101.101?[confirm]
Building configuration...[OK]
Connected to 172.16.101.101
Device#
```

Storing a Startup Configuration File on an FTP Server

The following example shows how to store a startup configuration file on a server by using FTP to copy the file:

```
Device# configure terminal

Device(config)# ip ftp username netadmin2

Device(config)# ip ftp password mypass

Device(config)# end

Device# copy nvram:startup-config ftp:

Remote host[]? 172.16.101.101

Name of configuration file to write [start-confg]?
Write file start-confg on host 172.16.101.101?[confirm]
![OK]
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from a TFTP Server to the Device

To copy a configuration file from a TFTP server to the device, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**copy tftp:** [[[*//location*]/*directory*]/*filename*] **system:running-config** **Example:** `Device# copy tftp://server1/dir10/datasource system:running-config`	Copies a configuration file from a TFTP server to the running configuration.
Step 3	**copy tftp:** [[[*//location*]/*directory*]/*filename*] **nvram:startup-config** **Example:** `Device# copy tftp://server1/dir10/datasource nvram:startup-config`	Copies a configuration file from a TFTP server to the startup configuration.
Step 4	**copy tftp:** [[[*//location*]/*directory*]/*filename*] **flash-[n]:/directory/startup-config** **Example:** `Device# copy tftp://server1/dir10/datasource flash:startup-config`	Copies a configuration file from a TFTP server to the startup configuration.

Examples

In the following example, the software is configured from the file named **tokyo-confg** at IP address 172.16.2.155:

```
Device# copy tftp://172.16.2.155/tokyo-confg system:running-config

Configure using tokyo-confg from 172.16.2.155? [confirm] Y

Booting tokyo-confg from 172.16.2.155:!!! [OK - 874/16000 bytes]
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from the rcp Server to the Device

To copy a configuration file from an rcp server to the running configuration or startup configuration, complete the following tasks:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Enters configuration mode from the terminal. This step is required only if you override the default remote username (see Step 3).
Step 3	**ip rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username NetAdmin1`	(Optional) Specifies the remote username.
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode. This step is required only if you override the default remote username (see Step 2).
Step 5	Do one of the following: • **copy rcp://**[*username*@]*location*[*directory*]*filename* **system:running-config** • **copy rcp://**[*username*@]*location*[*directory*]*filename* **nvram:startup-config** **Example:** `Device# copy rcp://[user1@example.com/dir10/fileone] nvram:startup-config`	Copies the configuration file from an rcp server to the running configuration or startup configuration.

Examples

Copy RCP Running-Config

The following example copies a configuration file named host1-confg from the netadmin1 directory on the remote server with an IP address of 172.16.101.101, and loads and runs the commands on the device:

```
device# copy rcp://netadmin1@172.16.101.101/host1-confg system:running-config
Configure using host1-confg from 172.16.101.101? [confirm]
Connected to 172.16.101.101
Loading 1112 byte file host1-confg:![OK]
device#
%SYS-5-CONFIG: Configured from host1-config by rcp from 172.16.101.101
```

Copy RCP Startup-Config

The following example specifies a remote username of netadmin1. Then it copies the configuration file named host2-confg from the netadmin1 directory on the remote server with an IP address of 172.16.101.101 to the startup configuration.

```
device# configure terminal
device(config)# ip rcmd remote-username netadmin1
device(config)# end
device# copy rcp: nvram:startup-config
Address of remote host [255.255.255.255]? 172.16.101.101
Name of configuration file[rtr2-confg]? host2-confg
Configure using host2-confg from 172.16.101.101?[confirm]
Connected to 172.16.101.101
Loading 1112 byte file host2-confg:![OK]
[OK]
device#
%SYS-5-CONFIG_NV:Non-volatile store configured from host2-config by rcp from 172.16.101.101
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from an FTP Server to the Device

To copy a configuration file from an FTP server to the running configuration or startup configuration, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Allows you to enter global configuration mode. This step is required only if you want to override the default remote username or password (see Steps 3 and 4).
Step 3	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username NetAdmin1`	(Optional) Specifies the default remote username.
Step 4	**ip ftp password** *password* **Example:**	(Optional) Specifies the default password.

	Command or Action	Purpose
	`Device(config)# ip ftp password adminpassword`	
Step 5	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode. This step is required only if you override the default remote username or password (see Steps 3 and 4).
Step 6	Do one of the following: • **copy ftp:** [[[//[*username*[**:***password*]**@**]*location*] **/***directory*]**/***filename*]**system:running-config** • **copy ftp** [[[**/***username*[*password*]**@**]*location*]**/***directory*]**/***filename*]**nvram:startup-config** **Example:** `Device# copy ftp:nvram:startup-config`	Using FTP copies the configuration file from a network server to running memory or the startup configuration.

Examples

Copy FTP Running-Config

The following example copies a host configuration file named host1-confg from the netadmin1 directory on the remote server with an IP address of 172.16.101.101, and loads and runs the commands on the device:

```
device# copy ftp://netadmin1:mypass@172.16.101.101/host1-confg system:running-config
Configure using host1-confg from 172.16.101.101? [confirm]
Connected to 172.16.101.101
Loading 1112 byte file host1-confg:![OK]
device#
%SYS-5-CONFIG: Configured from host1-config by ftp from 172.16.101.101
```

Copy FTP Startup-Config

The following example specifies a remote username of netadmin1. Then it copies the configuration file named host2-confg from the netadmin1 directory on the remote server with an IP address of 172.16.101.101 to the startup configuration:

```
device# configure terminal
device(config)# ip ftp username netadmin1
device(config)# ip ftp password mypass
device(config)# end
device# copy ftp: nvram:startup-config
Address of remote host [255.255.255.255]? 172.16.101.101
Name of configuration file[host1-confg]? host2-confg
Configure using host2-confg from 172.16.101.101?[confirm]
Connected to 172.16.101.101
Loading 1112 byte file host2-confg:![OK]
[OK]
device#
```

```
%SYS-5-CONFIG_NV:Non-volatile store configured from host2-config by ftp from 172.16.101.101
```

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Maintaining Configuration Files Larger than NVRAM

To maintain a configuration file that exceeds the size of NVRAM, perform the tasks described in the following sections:

Compressing the Configuration File

To compress configuration files, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service compress-config** **Example:** `Device(config)# service compress-config`	Specifies that the configuration file be compressed.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 5	Do one of the following: • Use FTP, RCP, or TFTP to copy the new configuration. • **configure terminal** **Example:** `Device# configure terminal`	Enters the new configuration: • If you try to load a configuration that is more than three times larger than the NVRAM size, the following error message is displayed: "[buffer overflow - *file-size* /*buffer-size* bytes]."

	Command or Action	Purpose
Step 6	**copy system:running-config nvram:startup-config** **Example:** `Device(config)# copy system:running-config nvram:startup-config`	When you have finished changing the running-configuration, save the new configuration.

Examples

The following example compresses a 129-KB configuration file to 11 KB:

```
Device# configure terminal

Device(config)# service compress-config

Device(config)# end

Device# copy tftp://172.16.2.15/tokyo-confg system:running-config

Configure using tokyo-confg from 172.16.2.155? [confirm] y

Booting tokyo-confg from 172.16.2.155:!!! [OK - 874/16000 bytes]
Device# copy system:running-config nvram:startup-config

Building configuration...
Compressing configuration from 129648 bytes to 11077 bytes
[OK]
```

Storing the Configuration in Flash Memory on Class A Flash File Systems

To store the startup configuration in flash memory, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy nvram:startup-config** *flash-filesystem:filename* **Example:** `Device# copy nvram:startup-config usbflash0:switch-config`	Copies the current startup configuration to the new location to create the configuration file.

	Command or Action	Purpose
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**boot config flash-filesystem:** *filename* **Example:** `Device(config)# boot config usbflash0:switch-config`	Specifies that the startup configuration file be stored in flash memory by setting the CONFIG_FILE variable.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 6	Do one of the following: • Use FTP, RCP, or TFTP to copy the new configuration. If you try to load a configuration that is more than three times larger than the NVRAM size, the following error message is displayed: "[buffer overflow - *file-size* /*buffer-size* bytes]. " • **configure terminal** **Example:** `Device# configure terminal`	Enters the new configuration.
Step 7	**copy system:running-config nvram:startup-config** **Example:** `Device(config)# copy system:running-config nvram:startup-config`	When you have finished changing the running-configuration, save the new configuration.

Examples

The following example stores the configuration file in usbflash0:

```
Device# copy nvram:startup-config usbflash0:switch-config

Device# configure terminal

Device(config)# boot config usbflash0:switch-config

Device(config)# end
```

```
Device# copy system:running-config nvram:startup-config
```

Loading the Configuration Commands from the Network

To use a network server to store large configurations, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy system:running-config** {**ftp:** | **rcp:** | **tftp:**} **Example:** `Device# copy system:running-config ftp:`	Saves the running configuration to an FTP, RCP, or TFTP server.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**boot network** {**ftp:**[[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] | **rcp:**[[[//[*username***@**]*location*]/*directory*]/*filename*] | **tftp:**[[[//*location*]/*directory*]/*filename*]} **Example:** `Device(config)# boot network ftp://user1:guessme@example.com/dir10/file1`	Specifies that the startup configuration file be loaded from the network server at startup.
Step 5	**service config** **Example:** `Device(config)# service config`	Enables the switch to download configuration files at system startup.
Step 6	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 7	**copy system:running-config nvram:startup-config**	Saves the configuration.

	Command or Action	Purpose
	Example: `Device# copy system:running-config nvram:startup-config`	

Copying Configuration Files from Flash Memory to the Startup or Running Configuration

To copy a configuration file from flash memory directly to your startup configuration in NVRAM or your running configuration, enter one of the commands in Step 2:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Do one of the following: • **copy** *filesystem*: [*partition-number***:**][*filename*] **nvram:startup-config** • **copy** *filesystem*: [*partition-number***:**][*filename*] **system:running-config** **Example:** `Device# copy usbflash0:4:ios-upgrade-1 nvram:startup-config`	• Loads a configuration file directly into NVRAM or • Copies a configuration file to your running configuration

Examples

The following example copies the file named ios-upgrade-1 from partition 4 of the flash memory PC Card in usbflash0 to the device startup configurations:

```
Device# copy usbflash0:4:ios-upgrade-1 nvram:startup-config

Copy 'ios-upgrade-1' from flash device as 'startup-config' ? [yes/no] yes

[OK]
```

Copying Configuration Files Between Flash Memory File Systems

On platforms with multiple flash memory file systems, you can copy files from one flash memory file system, such as internal flash memory to another flash memory file system. Copying files to different flash memory file systems lets you create backup copies of working configurations and duplicate configurations for other devices. To copy a configuration file between flash memory file systems, use the following commands in EXEC mode:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show** *source-filesystem*: **Example:** `Device# show flash:`	Displays the layout and contents of flash memory to verify the filename.
Step 3	**copy** *source-filesystem*:[*partition-number*:][*filename*] *dest-filesystem*:[*partition-number*:][*filename*] **Example:** `Device# copy flash: usbflash0:`	Copies a configuration file between flash memory devices. • The source device and the destination device cannot be the same. For example, the **copy usbflash0: usbflash0:** command is invalid.

Example

The following example copies the file named running-config from partition 1 on internal flash memory to partition 1 of usbflash0 on a device. In this example, the source partition is not specified, so the device prompts for the partition number:

```
Device# copy flash: usbflash0:

System flash
Partition   Size    Used      Free      Bank-Size  State          Copy Mode
  1         4096K   3070K     1025K     4096K      Read/Write     Direct
  2        16384K   1671K    14712K     8192K      Read/Write     Direct
[Type ?<no> for partition directory; ? for full directory; q to abort]
Which partition? [default = 1]
System flash directory, partition 1:
File  Length   Name/status
  1   3142748  dirt/network/mars-test/c3600-j-mz.latest
  2   850      running-config
[3143728 bytes used, 1050576 available, 4194304 total]
usbflash0 flash directory:
File  Length   Name/status
  1   1711088  dirt/gate/c3600-i-mz
```

```
  2   850      running-config
[1712068 bytes used, 2482236 available, 4194304 total]
Source file name? running-config

Destination file name [running-config]?
Verifying checksum for 'running-config' (file # 2)...  OK
Erase flash device before writing? [confirm]
Flash contains files. Are you sure you want to erase? [confirm]
Copy 'running-config' from flash: device
  as 'running-config' into usbflash0: device WITH erase? [yes/no] yes

Erasing device... eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee ...erased!

 [OK - 850/4194304 bytes]
Flash device copy took 00:00:30 [hh:mm:ss]
Verifying checksum...  OK (0x16)
```

Copying a Configuration File from an FTP Server to Flash Memory Devices

To copy a configuration file from an FTP server to a flash memory device, complete the task in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Enters global configuration mode. This step is required only if you override the default remote username or password (see Steps 3 and 4).
Step 3	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username Admin01`	(Optional) Specifies the remote username.
Step 4	**ip ftp password** *password* **Example:** `Device(config)# ip ftp password adminpassword`	(Optional) Specifies the remote password.
Step 5	**end** **Example:** `Device(config)# end`	(Optional) Exits configuration mode. This step is required only if you override the default remote username (see Steps 3 and 4).

	Command or Action	Purpose
Step 6	**copy ftp:** [[*//location*]*/directory*]*/bundle_name* **flash:** **Example:** `Device>copy ftp:/ie35xx-17.17.1.spa.bin flash:`	Copies the configuration file from a network server to the flash memory device using FTP.

What to Do Next

After you have issued the **copy** EXEC command, you may be prompted for additional information or for confirmation of the action. The prompt displayed depends on how much information you provide in the **copy** command and the current setting of the **file prompt** global configuration command.

Copying a Configuration File from an RCP Server to Flash Memory Devices

To copy a configuration file from an RCP server to a flash memory device, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	(Optional) Enters global configuration mode. This step is required only if you override the default remote username or password (see Step 3).
Step 3	**ip rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username Admin01`	(Optional) Specifies the remote username.
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Exits configuration mode. This step is required only if you override the default remote username or password (see Step 3).
Step 5	**copy rcp:** [[[*//*[*username@*]*location*]*/directory*] */bundle_name*] **flash:** **Example:** `Device# copy`	Copies the configuration file from a network server to the flash memory device using RCP. Respond to any device prompts for additional information or confirmation. Prompting depends on how much information you provide in the

	Command or Action	Purpose
	`rcp://netadmin@172.16.101.101/bundle1 flash:`	**copy** command and the current setting of the **file prompt** command.

Copying a Configuration File from a TFTP Server to Flash Memory Devices

To copy a configuration file from a TFTP server to a flash memory device, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy tftp:** [[[*//location*]*/directory*]/bundle_name **flash:** **Example:** `Device# copy tftp://ie35xx_caa-universalk9.SSA.03.12.02.EZP.150-12.02.EZP.150-12.02.EZP.bin flash:`	Copies the file from a TFTP server to the flash memory device. Reply to any device prompts for additional information or confirmation. Prompting depends on how much information you provide in the **copy** command and the current setting of the **file prompt** command.

Examples

The following example shows the copying of the configuration file named switch-config from a TFTP server to the flash memory card inserted in usbflash0. The copied file is renamed new-config.

```
Device#
copy tftp:switch-config usbflash0:new-config
```

Re-executing the Configuration Commands in the Startup Configuration File

To re-execute the commands located in the startup configuration file, complete the task in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure memory** **Example:** `Device# configure memory`	Re-executes the configuration commands located in the startup configuration file.

Clearing the Startup Configuration

You can clear the configuration information from the startup configuration. If you reboot the device with no startup configuration, the device enters the Setup command facility so that you can configure the device from scratch. To clear the contents of your startup configuration, complete the task in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**erase nvram** **Example:** `Device# erase nvram`	Clears the contents of your startup configuration. **Note** For all platforms except the Class A Flash file system platforms, this command erases NVRAM. The startup configuration file cannot be restored once it has been deleted. On Class A Flash file system platforms, when you use the **erase startup-config** EXEC command, the device erases or deletes the configuration pointed to by the CONFIG_FILE environment variable. If this variable points to NVRAM, the device erases NVRAM. If the CONFIG_FILE environment variable specifies a flash memory device and configuration filename, the device deletes the configuration file. That is, the device marks the file as "deleted," rather than erasing it. This feature allows you to recover a deleted file.

Deleting a Specified Configuration File

To delete a specified configuration on a specific flash device, complete the task in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**delete** *flash-filesystem***:***filename* **Example:** `Device# delete usbflash0:myconfig`	Deletes the specified configuration file on the specified flash device. **Note** On Class A and B Flash file systems, when you delete a specific file in flash memory, the system marks the file as deleted, allowing you to later recover a deleted file using the **undelete** EXEC command. Erased files cannot be recovered. To permanently erase the configuration file, use the **squeeze** EXEC command. On Class C Flash file systems, you cannot recover a file that has been deleted. If you attempt to erase or delete the configuration file specified by the CONFIG_FILE environment variable, the system prompts you to confirm the deletion.

Specifying the CONFIG_FILE Environment Variable on Class A Flash File Systems

On Class A flash file systems, you can configure the Cisco IOS software to load the startup configuration file specified by the CONFIG_FILE environment variable. The CONFIG_FILE variable defaults to NVRAM. To change the CONFIG_FILE environment variable, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy** [*flash-url* | *ftp-url* | *rcp-url* | *tftp-url* | **system:running-config** | **nvram:startup-config**] *dest-flash-url* **Example:**	Copies the configuration file to the flash file system from which the device loads the file on restart.

	Command or Action	Purpose
	`Device# copy system:running-config nvram:startup-config`	
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**boot config** *dest-flash-url* **Example:** `Device(config)# boot config 172.16.1.1`	Sets the CONFIG_FILE environment variable. This step modifies the runtime CONFIG_FILE environment variable.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 6	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the configuration performed in Step 3 to the startup configuration.
Step 7	**show boot** **Example:** `Device# show boot`	(Optional) Allows you to verify the contents of the CONFIG_FILE environment variable.

Examples

The following example copies the running configuration file to the device. This configuration is then used as the startup configuration when the system is restarted:

```
Device# copy system:running-config usbflash0:config2
Device# configure terminal
Device(config)# boot config usbflash0:config2
Device(config)# end
Device# copy system:running-config nvram:startup-config
[ok]
Device# show boot
BOOT variable = usbflash0:rsp-boot-m
CONFIG_FILE variable = nvram:
Current CONFIG_FILE variable = usbflash0:config2
Configuration register is 0x010F
```

What to Do Next

After you specify a location for the startup configuration file, the **nvram:startup-config** command is aliased to the new location of the startup configuration file. The **more nvram:startup-config** EXEC command displays the startup configuration, regardless of its location. The **erase nvram:startup-config** EXEC command erases the contents of NVRAM and deletes the file pointed to by the CONFIG_FILE environment variable.

When you save the configuration using the **copy system:running-config nvram:startup-config** command, the device saves a complete version of the configuration file to the location specified by the CONFIG_FILE environment variable and a distilled version to NVRAM. A distilled version is one that does not contain access list information. If NVRAM contains a complete configuration file, the device prompts you to confirm your overwrite of the complete version with the distilled version. If NVRAM contains a distilled configuration, the device does not prompt you for confirmation and proceeds with overwriting the existing distilled configuration file in NVRAM.

Note If you specify a file in a flash device as the CONFIG_FILE environment variable, every time you save your configuration file with the **copy system:running-config nvram:startup-config** command, the old configuration file is marked as "deleted," and the new configuration file is saved to that device. Eventually, Flash memory fills up as the old configuration files still take up memory. Use the **squeeze** EXEC command to permanently delete the old configuration files and reclaim the space.

Configuring the Device to Download Configuration Files

You can specify an ordered list of network configuration and host configuration filenames. The Cisco IOS XE software scans this list until it loads the appropriate network or host configuration file.

To configure the device to download configuration files at system startup, perform at least one of the tasks described in the following sections:

- "Configuring the Switch to Download the Network Configuration File"
- "Configuring the Switch to Download the Network Configuration File"

If the device fails to load a configuration file during startup, it tries again every 10 minutes (the default setting) until a host provides the requested files. With each failed attempt, the device displays the following message on the console terminal:

```
Booting host-confg... [timed out]
```

If there are any problems with the startup configuration file, or if the configuration register is set to ignore NVRAM, the device enters the Setup command facility.

Configuring the Device to Download the Network Configuration File

To configure the Cisco IOS software to download a network configuration file from a server at startup, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**boot network** {**ftp:**[[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] | **rcp:**[[[//[*username***@**]*location*]/*directory*]/*filename*] | **tftp:**[[[//*location*]/*directory*]/*filename*]} **Example:** `Device(config)# boot network tftp:hostfile1`	Specifies the network configuration file to download at startup, and the protocol to be used (TFTP, RCP, or FTP). • If you do not specify a network configuration filename, the Cisco IOS software uses the default filename network-confg. If you omit the address, the device uses the broadcast address. • You can specify more than one network configuration file. The software tries them in order entered until it loads one. This procedure can be useful for keeping files with different configuration information loaded on a network server.
Step 4	**service config** **Example:** `Device(config)# service config`	Enables the system to automatically load the network file on restart.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 6	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the running configuration to the startup configuration file.

Configuring the Device to Download the Host Configuration File

To configure the Cisco IOS software to download a host configuration file from a server at startup, complete the tasks in this section:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**boot host** {**ftp:**[[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] | **rcp:**[[[//[*username***@**]*location*]/*directory*]/*filename*] | **tftp:**[[[//*location*]/*directory*]/*filename*] } **Example:** `Device(config)# boot host tftp:hostfile1`	Specifies the host configuration file to download at startup, and the protocol to be used (FTP, RCP, or TFTP): • If you do not specify a host configuration filename, the device uses its own name to form a host configuration filename by converting the name to all lowercase letters, removing all domain information, and appending "-confg." If no host name information is available, the software uses the default host configuration filename device-confg. If you omit the address, the device uses the broadcast address. • You can specify more than one host configuration file. The Cisco IOS software tries them in order entered until it loads one. This procedure can be useful for keeping files with different configuration information loaded on a network server.
Step 4	**service config** **Example:** `Device(config)# service config`	Enables the system to automatically load the host file upon restart.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.

	Command or Action	Purpose
Step 6	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the running configuration to the startup configuration file.

Example

In the following example, a device is configured to download the host configuration file named hostfile1 and the network configuration file named networkfile1. The device uses TFTP and the broadcast address to obtain the file:

```
Device# configure terminal
Device(config)# boot host tftp:hostfile1
Device(config)# boot network tftp:networkfile1
Device(config)# service config
Device(config)# end
Device# copy system:running-config nvram:startup-config
```

CHAPTER 142

Secure Copy

This document provides the procedure to configure a Cisco device for Secure Copy (SCP) server-side functionality.

Prerequisites for Secure Copy

- Configure Secure Shell (SSH), authentication, and authorization on the device.
- Because the Secure Copy Protocol (SCP) relies on SSH for its secure transport, the device must have a Rivest, Shamir, and Adelman (RSA) key pair.

Information About Secure Copy

The Secure Copy feature provides a secure and authenticated method for copying switch configurations or switch image files. The Secure Copy Protocol (SCP) relies on Secure Shell (SSH), an application and a protocol that provides a secure replacement for the Berkeley r-tools.

The behavior of SCP is similar to that of Remote Copy Protocol (RCP), which comes from the Berkeley r-tools suite (Berkeley university's own set of networking applications), except that SCP relies on SSH for security. In addition, SCP requires authentication, authorization, and accounting (AAA) to be configured to ensure that the device can determine whether a user has the correct privilege level.

SCP allows only users with a privilege level of 15 to copy a file in the Cisco IOS File System (Cisco IFS) to and from a device by using the **copy** command. An authorized administrator can also perform this action from a workstation.

Note

- Enable the SCP option while using the `pscp.exe` file.
- An RSA public-private key pair must be configured on the device for SSH to work.

Similar to SCP, SSH File Transfer Protocol (SFTP) can be used to copy switch configuration or image files. For more information, refer the *Configuring SSH File Transfer Protocol* chapter of the *Security Configuration Guide*.

Secure Copy Performance Improvements

SSH bulk data transfer mode can be used to enhance the throughput performance of SCP that is operating in the capacity of a client or a server. SSH bulk data transfer mode is enabled by default with default window size of 128KB. TCP selective acknowledgement (SACK) is enabled by default if the bulk mode window size is configured.

The default bulk mode window size of 128 KB is optimal to copy large files in most network settings. However, in long big networks where the round-trip time (RTT) is high, 128 KB is not enough. You can enable the most optimal SCP throughput performance by configuring the bulk mode window size using the **ip ssh bulk-mode** *window-size* command. For example, in an ideal lab testing environment, a window size of 2 MB in a 200-milliseconds round-trip time setting can give around 500 percent improved throughput performance when compared to the default 128-KB window size.

The bulk mode window size must be configured as per the network bandwidth-delay product, that is, a multiple of total available bandwidth in bits per second and the round-trip time in seconds. Because the CPU usage may increase with the increased window size, make sure to balance this by choosing the right window size.

How to Configure Secure Copy

The following sections provide information about the Secure Copy configuration tasks.

Configuring Secure Copy

To configure a Cisco device for SCP server-side functionality, perform the following steps.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Sets AAA authentication at login.

	Command or Action	Purpose
Step 4	**aaa authentication login** {**default** | *list-name*} *method1* [*method2...*] **Example:** `Device(config)# aaa authentication login default group tacacs+`	Enables the AAA access control system.
Step 5	**username** *name* [**privilege** *level*] **password** *encryption-type encrypted-password* **Example:** `Device(config)# username superuser privilege 2 password 0 superpassword`	Establishes a username-based authentication system. **Note** You can omit this step if a network-based authentication mechanism, such as TACACS+ or RADIUS, has been configured.
Step 6	**ip scp server enable** **Example:** `Device(config)# ip scp server enable`	Enables SCP server-side functionality.
Step 7	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 8	**debug ip scp** **Example:** `Device# debug ip scp`	(Optional) Troubleshoots SCP authentication problems.

Configuring SCP Username Password

To configure a username and password for SCP, perform the following steps:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip scp username***username* **Example:** `Device# ip scp username cisco`	Defines the username.
Step 4	**ip scp password***password* **Example:** `Device# ip scp password 0 cisco`	Defines the password. Specify the encryption level. • 0 – Unencrypted password. • 0 – Encrypted password. • Line – Clear text password.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Enabling Secure Copy on the SSH Server

The following task shows how to configure the server-side functionality for SCP. This task shows a typical configuration that allows a device to securely copy files from a remote workstation.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables the Authentication, Authorization, and Accounting (AAA) access control model.
Step 4	**aaa authentication login default local** **Example:** `Device(config)# aaa authentication login default local`	Sets AAA authentication to use the local username database for authentication at login.
Step 5	**aaa authorization exec default local** **Example:**	Sets the parameters that restrict user access to a network, runs the authorization to determine if the user ID is allowed to run an privileged

	Command or Action	Purpose
	`Device(config)# aaa authorization exec default local`	EXEC shell, and specifies that the system must use the local database for authorization.
Step 6	**username** *name* **privilege** *privilege-level* **password** *password* **Example:** `Device(config)# username samplename privilege 15 password password1`	Establishes a username-based authentication system, and specifies the username, privilege level, and an unencrypted password. **Note** The minimum required value for the *privilege-level* argument is 15. A privilege level of less than 15 results in the connection closing.
Step 7	**ip ssh time-out** *seconds* **Example:** `Device(config)# ip ssh time-out 120`	Sets the time interval (in seconds) that the device waits for the SSH client to respond.
Step 8	**ip ssh authentication-retries** *integer* **Example:** `Device(config)# ip ssh authentication-retries 3`	Sets the number of authentication attempts after which the interface is reset.
Step 9	**ip scp server enable** **Example:** `Device(config)# ip scp server enable`	Enables the device to securely copy files from a remote workstation.
Step 10	**ip ssh bulk-mode** *window-size* **Example:** `Device(config)# ip ssh bulk-mode 33107232`	(Optional) Sets the bulk mode window size to enhance the throughput performance of SCP. **Note** SSH bulk data transfer mode is enabled by default with default window size of 128KB.
Step 11	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 12	**debug ip scp** **Example:** `Device# debug ip scp`	(Optional) Provides diagnostic information about SCP authentication problems.

Configuration Examples for Secure Copy

The following are examples of the Secure Copy configuration.

Example: Secure Copy Configuration Using Local Authentication

The following example shows how to configure the server-side functionality of Secure Copy. This example uses a locally defined username and password.

```
! AAA authentication and authorization must be configured properly in order for SCP to work.
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication login default local
Device(config)# aaa authorization exec default local
Device(config)# username user1 privilege 15 password 0 lab
! SSH must be configured and functioning properly.
Device(config)# ip scp server enable
Device(config)# end
```

Example: Secure Copy Server-Side Configuration Using Network-Based Authentication

The following example shows how to configure the server-side functionality of Secure Copy using a network-based authentication mechanism:

```
! AAA authentication and authorization must be configured properly for SCP to work.
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication login default group tacacs+
Device(config)# aaa authorization exec default group tacacs+
! SSH must be configured and functioning properly.
Device(config)# ip ssh time-out 120
Device(config)# ip ssh authentication-retries 3
Device(config)# ip scp server enable
Device(config)# end
```

CHAPTER 143

Configuration Replace and Configuration Rollback

Prerequisites for Configuration Replace and Configuration Rollback

The format of the configuration files used as input by the Configuration Replace and Configuration Rollback feature must comply with standard Cisco software configuration file indentation rules as follows:

- Start all commands on a new line with no indentation, unless the command is within a configuration submode.
- Indent commands within a first-level configuration submode one space.
- Indent commands within a second-level configuration submode two spaces.
- Indent commands within subsequent submodes accordingly.

These indentation rules describe how the software creates configuration files for such commands as **show running-config** or **copy running-config** *destination-url*. Any configuration file generated on a Cisco device complies with these rules.

Free memory larger than the combined size of the two configuration files (the current running configuration and the saved replacement configuration) is required.

Restrictions for Configuration Replace and Configuration Rollback

If the device does not have free memory larger than the combined size of the two configuration files (the current running configuration and the saved replacement configuration), the configuration replace operation is not performed.

Certain Cisco configuration commands such as those pertaining to physical components of a networking device (for example, physical interfaces) cannot be added or removed from the running configuration. For example, a configuration replace operation cannot remove the **interface gigabitethernet 1/1** command line from the current running configuration if that interface is physically present on the device. Similarly, the **interface gigabitethernet 1/3** command line cannot be added to the running configuration if no such interface is physically present on the device. A configuration replace operation that attempts to perform these types of changes results in error messages indicating that these specific command lines failed.

In very rare cases, certain Cisco configuration commands cannot be removed from the running configuration without reloading the device. A configuration replace operation that attempts to remove this type of command results in error messages indicating that these specific command lines failed.

Information About Configuration Replace and Configuration Rollback

Configuration Archive

The Cisco IOS configuration archive is intended to provide a mechanism to store, organize, and manage an archive of Cisco IOS configuration files to enhance the configuration rollback capability provided by the **configure replace** command. Before this feature was introduced, you could save copies of the running configuration using the **copy running-config** *destination-url* command, storing the replacement file either locally or remotely. However, this method lacked any automated file management. On the other hand, the Configuration Replace and Configuration Rollback feature provides the capability to automatically save copies of the running configuration to the Cisco IOS configuration archive. These archived files serve as checkpoint configuration references and can be used by the **configure replace** command to revert to previous configuration states.

The **archive config** command allows you to save Cisco IOS configurations in the configuration archive using a standard location and filename prefix that is automatically appended with an incremental version number (and optional timestamp) as each consecutive file is saved. This functionality provides a means for consistent identification of saved Cisco IOS configuration files. You can specify how many versions of the running configuration are kept in the archive. After the maximum number of files are saved in the archive, the oldest file is automatically deleted when the next, most recent file is saved. The **show archive** command displays information for all configuration files saved in the Cisco IOS configuration archive.

The Cisco IOS configuration archive, in which the configuration files are stored and available for use with the **configure replace** command, can be located on the following file systems: FTP, HTTP, RCP, TFTP.

Configuration Replace

The **configure replace** privileged EXEC command provides the capability to replace the current running configuration with any saved Cisco IOS configuration file. This functionality can be used to revert to a previous configuration state, effectively rolling back any configuration changes that were made since the previous configuration state was saved.

When using the **configure replace** command, you must specify a saved Cisco IOS configuration as the replacement configuration file for the current running configuration. The replacement file must be a complete configuration generated by a Cisco IOS device (for example, a configuration generated by the **copy running-config** *destination-url* command), or, if generated externally, the replacement file must comply with the format of files generated by Cisco IOS devices. When the **configure replace** command is entered, the current running configuration is compared with the specified replacement configuration and a set of diffs is generated. The algorithm used to compare the two files is the same as that employed by the **show archive config differences** command. The resulting diffs are then applied by the Cisco IOS parser to achieve the replacement configuration state. Only the diffs are applied, avoiding potential service disruption from reapplying configuration commands that already exist in the current running configuration. This algorithm effectively handles configuration changes to order-dependent commands (such as access lists) through a multiple pass process. Under normal circumstances, no more than three passes are needed to complete a configuration replace operation, and a limit of five passes is performed to preclude any looping behavior.

The Cisco IOS **copy** *source-url* **running-config** privileged EXEC command is often used to copy a stored Cisco IOS configuration file to the running configuration. When using the **copy** *source-url* **running-config** command as an alternative to the **configure replace** *target-url* privileged EXEC command, the following major differences should be noted:

- The **copy** *source-url* **running-config** command is a merge operation and preserves all of the commands from both the source file and the current running configuration. This command does not remove commands from the current running configuration that are not present in the source file. In contrast, the **configure replace** *target-url* command removes commands from the current running configuration that are not present in the replacement file and adds commands to the current running configuration that need to be added.
- The **copy** *source-url* **running-config** command applies every command in the source file, whether or not the command is already present in the current running configuration. This algorithm is inefficient and, in some cases, can result in service outages. In contrast, the **configure replace** *target-url* command only applies the commands that need to be applied—no existing commands in the current running configuration are reapplied.
- A partial configuration file may be used as the source file for the **copy** *source-url* **running-config** command, whereas a complete Cisco IOS configuration file must be used as the replacement file for the **configure replace** *target-url* command.

A locking feature for the configuration replace operation was introduced. When the **configure replace** command is used, the running configuration file is locked by default for the duration of the configuration replace operation. This locking mechanism prevents other users from changing the running configuration while the replacement operation is taking place, which might otherwise cause the replacement operation to terminate unsuccessfully. You can disable the locking of the running configuration by using the **no lock** keyword when issuing the **configure replace** command.

The running configuration lock is automatically cleared at the end of the configuration replace operation. You can display any locks that may be currently applied to the running configuration using the **show configuration lock** command.

Configuration Rollback

The concept of rollback comes from the transactional processing model common to database operations. In a database transaction, you might make a set of changes to a given database table. You then must choose whether to commit the changes (apply the changes permanently) or to roll back the changes (discard the changes and revert to the previous state of the table). In this context, rollback means that a journal file containing a log of the changes is discarded, and no changes are applied. The result of the rollback operation is to revert to the previous state, before any changes were applied.

The **configure replace** command allows you to revert to a previous configuration state, effectively rolling back changes that were made since the previous configuration state was saved. Instead of basing the rollback operation on a specific set of changes that were applied, the Cisco IOS configuration rollback capability uses the concept of reverting to a specific configuration state based on a saved Cisco IOS configuration file. This concept is similar to the database idea of saving a checkpoint (a saved version of the database) to preserve a specific state.

If the configuration rollback capability is desired, you must save the Cisco IOS running configuration before making any configuration changes. Then, after entering configuration changes, you can use that saved configuration file to roll back the changes (using the **configure replace** *target-url* command). Furthermore, because you can specify any saved Cisco IOS configuration file as the replacement configuration, you are not limited to a fixed number of rollbacks, as is the case in some rollback models.

Configuration Rollback Confirmed Change

The Configuration Rollback Confirmed Change feature allows configuration changes to be performed with an optional requirement that they be confirmed. If this confirmation is not received, the configuration is returned to the state prior to the changes being applied. The mechanism provides a safeguard against inadvertent loss of connectivity between a network device and the user or management application due to configuration changes.

Benefits of Configuration Replace and Configuration Rollback

- Allows you to revert to a previous configuration state, effectively rolling back configuration changes.
- Allows you to replace the current running configuration file with the startup configuration file without having to reload the device or manually undo CLI changes to the running configuration file, therefore reducing system downtime.
- Allows you to revert to any saved Cisco IOS configuration state.
- Simplifies configuration changes by allowing you to apply a complete configuration file to the device, where only the commands that need to be added or removed are affected.
- When using the **configure replace** command as an alternative to the **copy** *source-url* **running-config** command, increases efficiency and prevents risk of service outages by not reapplying existing commands in the current running configuration.

How to Use Configuration Replace and Configuration Rollback

Creating a Configuration Archive

No prerequisite configuration is needed to use the **configure replace** command. Using the **configure replace** command in conjunction with the Cisco IOS configuration archive and the **archive config** command is optional but offers significant benefit for configuration rollback scenarios. Before using the **archive config** command, the configuration archive must be configured. Perform this task to configure the characteristics of the configuration archive.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**archive** **Example:** `Device(config)# archive`	Enters archive configuration mode.
Step 4	**path** *url* **Example:** `Device(config-archive)# path flash:myconfiguration`	Specifies the location and filename prefix for the files in the Cisco IOS configuration archive. **Note** If a directory is specified in the path instead of file, the directory name must be followed by a forward slash as follows: path flash:/directory/. The forward slash is not necessary after a filename; it is only necessary when specifying a directory.
Step 5	**maximum** *number* **Example:** `Device(config-archive)# maximum 14`	(Optional) Sets the maximum number of archive files of the running configuration to be saved in the Cisco IOS configuration archive. • The *number* argument is the maximum number of archive files of the running configuration to be saved in the Cisco IOS

	Command or Action	Purpose
		configuration archive. Valid values are from 1 to 14. The default is 10. **Note** Before using this command, you must configure the **path** command to specify the location and filename prefix for the files in the Cisco IOS configuration archive.
Step 6	**time-period** *minutes* **Example:** `Device(config-archive)# time-period 1440`	(Optional) Sets the time increment for automatically saving an archive file of the current running configuration in the Cisco IOS configuration archive. • The *minutes* argument specifies how often, in minutes, to automatically save an archive file of the current running configuration in the Cisco IOS configuration archive. **Note** Before using this command, you must configure the **path** command to specify the location and filename prefix for the files in the Cisco IOS configuration archive.
Step 7	**end** **Example:** `Device(config-archive)# end`	Exits to privileged EXEC mode.
Step 8	**archive config** **Example:** `Device# archive config`	Saves the current running configuration file to the configuration archive. **Note** The **path** command must be configured before using this command.

Performing a Configuration Replace or Configuration Rollback Operation

Perform this task to replace the current running configuration file with a saved Cisco IOS configuration file.

Note You must create a configuration archive before performing this procedure. See Creating a Configuration Archive for detailed steps. The following procedure details how to return to that archived configuration in the event of a problem with the current running configuration.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure replace** *target-url* [**nolock**] [**list**] [**force**] [**ignore case**] [**revert trigger** [**error**] [**timer** *minutes*] | **time** *minutes*]] **Example:** `Device# configure replace flash: startup-config time 120`	Replaces the current running configuration file with a saved Cisco IOS configuration file. • The *target - url* argument is a URL (accessible by the Cisco IOS file system) of the saved Cisco IOS configuration file that is to replace the current running configuration, such as the configuration file created using the **archive config** command. • The **list** keyword displays a list of the command lines applied by the Cisco IOS software parser during each pass of the configuration replace operation. The total number of passes performed is also displayed. • The **force** keyword replaces the current running configuration file with the specified saved Cisco IOS configuration file without prompting you for confirmation. • The **time** *minutes* keyword and argument specify the time (in minutes) within which you must enter the **configure confirm** command to confirm replacement of the current running configuration file. If the **configure confirm** command is not entered within the specified time limit, the configuration replace operation is automatically reversed (in other words, the current running configuration file is restored to the configuration state that existed prior to entering the **configure replace** command). • The **nolock** keyword disables the locking of the running configuration file that prevents other users from changing the running configuration during a configuration replace operation.

	Command or Action	Purpose
		• The **revert trigger** keywords set the following triggers for reverting to the original configuration: • **error**: Reverts to the original configuration upon error. • **timer** *minutes*: Reverts to the original configuration if specified time elapses. **Note** In some cases, while performing the **revert trigger** operation for multiple pass operations, a partial configuration may be missed out causing the revert operation to the original configuration state to fail. • The **ignore case** keyword allows the configuration to ignore the case of the confirmation command.
Step 3	**configure revert** { **now** | **timer** {*minutes* | **idle** *minutes*} } **Example:** `Device# configure revert now`	(Optional) To cancel the timed rollback and trigger the rollback immediately, or to reset parameters for the timed rollback, use the **configure revert**command in privileged EXEC mode. • **now**: Triggers the rollback immediately. • **timer**: Resets the configuration revert timer. • Use the *minutes* argument with the **timer** keyword to specify a new revert time in minutes. • Use the **idle** keyword along with a time in minutes to set the maximum allowable time period of no activity before reverting to the saved configuration.
Step 4	**configure confirm** **Example:** `Device# configure confirm`	(Optional) Confirms replacement of the current running configuration file with a saved Cisco IOS configuration file. **Note** Use this command only if the **time** *seconds* keyword and argument of the **configure replace** command are specified.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device# exit`	Exits to user EXEC mode.

Monitoring and Troubleshooting the Feature

Perform this task to monitor and troubleshoot the Configuration Replace and Configuration Rollback feature.

Procedure

Step 1 **enable**

Use this command to enable privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
Device#
```

Step 2 **show archive**

Use this command to display information about the files saved in the Cisco IOS configuration archive.

Example:

```
Device# show archive
There are currently 1 archive configurations saved.
The next archive file will be named flash:myconfiguration-2
 Archive #  Name
   0
   1       flash:myconfiguration-1 <- Most Recent
   2
   3
   4
   5
   6
   7
   8
   9
   10
   11
   12
   13
   14
```

The following is sample output from the **show archive** command after several archive files of the running configuration have been saved. In this example, the maximum number of archive files to be saved is set to three.

Example:

```
Device# show archive
```

```
There are currently 3 archive configurations saved.
The next archive file will be named flash:myconfiguration-8
 Archive #  Name
   0
   1       :Deleted
   2       :Deleted
   3       :Deleted
   4       :Deleted
   5       flash:myconfiguration-5
   6       flash:myconfiguration-6
   7       flash:myconfiguration-7 <- Most Recent
   8
   9
   10
   11
   12
   13
   14
```

Step 3 debug archive versioning

Use this command to enable debugging of the Cisco IOS configuration archive activities to help monitor and troubleshoot configuration replace and rollback.

Example:

```
Device# debug archive versioning
Jan  9 06:46:28.419:backup_running_config
Jan  9 06:46:28.419:Current = 7
Jan  9 06:46:28.443:Writing backup file flash:myconfiguration-7
Jan  9 06:46:29.547: backup worked
```

Step 4 debug archive config timestamp

Use this command to enable debugging of the processing time for each integral step of a configuration replace operation and the size of the configuration files being handled.

Example:

```
Device# debug archive config timestamp
Device# configure replace flash:myconfiguration force
Timing Debug Statistics for IOS Config Replace operation:
       Time to read file usbflash0:sample_2.cfg = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
Starting Pass 1
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:93
       Size of file        :2539
       Time taken for positive rollback pass = 320 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for negative incremental diffs pass = 59 msec (0 sec)
       Time taken by PI to apply changes = 0 msec (0 sec)
       Time taken for Pass 1 = 380 msec (0 sec)
Starting Pass 2
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
       Time taken for positive rollback pass = 0 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for Pass 2 = 0 msec (0 sec)
Total number of passes:1
Rollback Done
```

Step 5 **exit**

Use this command to exit to user EXEC mode.

Example:

```
Device# exit
Device>
```

Configuration Examples for Configuration Replace and Configuration Rollback

Creating a Configuration Archive

The following example shows how to perform the initial configuration of the Cisco IOS configuration archive. In this example, flash:myconfiguration is specified as the location and filename prefix for the files in the configuration archive and a value of 10 is set as the maximum number of archive files to be saved.

```
configure terminal
!
archive
 path flash:myconfiguration
 maximum 10
end
```

Replacing the Current Running Configuration with a Saved Cisco IOS Configuration File

The following example shows how to replace the current running configuration with a saved Cisco IOS configuration file named flash:myconfiguration. The **configure replace** command interactively prompts you to confirm the operation.

```
Device# configure replace flash:myconfiguration
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
Total number of passes: 1
Rollback Done
```

In the following example, the **list** keyword is specified in order to display the command lines that were applied during the configuration replace operation:

```
Device# configure replace flash:myconfiguration list
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
```

```
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
!Pass 1
!List of Commands:
no snmp-server community public ro
snmp-server community mystring ro

end
Total number of passes: 1
Rollback Done
```

Reverting to the Startup Configuration File

The following example shows how to revert to the Cisco IOS startup configuration file using the **configure replace** command. This example also shows the use of the optional **force** keyword to override the interactive user prompt:

```
Device# configure replace flash:startup-config force
Total number of passes: 1
Rollback Done
```

Performing a Configuration Replace Operation with the configure confirm Command

The following example shows the use of the **configure replace** command with the **time** *minutes* keyword and argument. You must enter the **configure confirm** command within the specified time limit to confirm replacement of the current running configuration file. If the **configure confirm** command is not entered within the specified time limit, the configuration replace operation is automatically reversed (in other words, the current running configuration file is restored to the configuration state that existed prior to entering the **configure replace** command).

```
Device# configure replace flash:startup-config time 120
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
Total number of passes: 1
Rollback Done
Device# configure confirm
```

The following example shows the use of the **configure revert** command with the **timer** keyword. You must enter the **configure revert** command to cancel the timed rollback and trigger the rollback immediately, or to reset parameters for the timed rollback.

```
Device# configure revert timer 100
```

Performing a Configuration Rollback Operation

The following example shows how to make changes to the current running configuration and then roll back the changes. As part of the configuration rollback operation, you must save the current running configuration before making changes to the file. In this example, the **archive config** command is used to save the current

running configuration. The generated output of the **configure replace** command indicates that only one pass was performed to complete the rollback operation.

Note Before using the **archive config** command, you must configure the **path** command to specify the location and filename prefix for the files in the Cisco IOS configuration archive.

You first save the current running configuration in the configuration archive as follows:

```
archive config
```

You then enter configuration changes as shown in the following example:

```
configure terminal
!
user netops2 password rain
user netops3 password snow
exit
```

After having made changes to the running configuration file, assume you now want to roll back these changes and revert to the configuration that existed before the changes were made. The **show archive** command is used to verify the version of the configuration to be used as a replacement file. The **configure replace** command is then used to revert to the replacement configuration file as shown in the following example:

```
Device# show archive
There are currently 1 archive configurations saved.
The next archive file will be named flash:myconfiguration-2
 Archive #  Name
   0
   1       flash:myconfiguration-1 <- Most Recent
   2
   3
   4
   5
   6
   7
   8
   9
   10
Device# configure replace flash:myconfiguration-1
Total number of passes: 1
Rollback Done
```

CHAPTER 144

Software Maintenance Upgrade

Software Maintenance Upgrade (a SMU), is a package that can be installed on a system to provide a fix or a security resolution to a released image.

Restrictions for Software Maintenance Upgrade

- Hot patching is not supported.
- SMU supports cold patching using install mode only.
- SMU installation will be supported in install mode only.

Information About Software Maintenance Upgrade

SMU Overview

An SMU is a package that can be installed on a system, to provide a fix or a security resolution to a released image. An SMU package is provided on a per release and per component basis.

An SMU provides a significant benefit over classic Cisco IOS software because it allows you to address network issues quickly while reducing the time and scope of the testing required. The Cisco IOS XE platform internally validates SMU compatibility and does not allow you to install incompatible SMUs.

All SMUs are integrated into the subsequent Cisco IOS XE software maintenance releases. An SMU is an independent and self-sufficient package and it does not have any prerequisites or dependencies. You can choose which SMUs to install or uninstall in any order.

SMUs are supported only on Extended Maintenance releases and for the full lifecycle of the underlying software release.

Perform these basic steps to install an SMU:

1. Add the SMU to the filesystem.

2. Activate the SMU on the system.
3. Commit the SMU changes so that it is persistent across reloads.

SMU Workflow

The SMU process is initiated with a request to the Cisco Customer Support. Contact your customer support to raise an SMU request.

At release time, the SMU package is posted to the Cisco Software Download page and can be downloaded and installed.

SMU Package

The SMU package contains a small set of files for patching the release along with metadata that describes the contents of the package, and fix for the reported issue that the SMU is requested for. The SMU package also supports patching of the public key infrastructure (PKI) component.

SMU Reload

All SMUs require a cold reload of the system during activation. A cold reload is the complete reload of the operating system. This action affects the traffic flow for the duration of the reload. This reload ensures that all processes are started with the correct libraries and files that are installed as part of the SMU.

Note If the user deletes the SMU file from the directory and performs a bootup, the device displays the error message **`%BOOT-3-BOOTTIME_SMU_MISSING_DETECTED: R0/0: install_engine: SMU file /bootflash/ie35xx-lni.17.17.1.SSA.bin missing and system impact will be unknown`**. However, this will not lead to any functional error.

How to Manage Software Maintenance Updates

The following sections provide information about managing SMUs.

You can install, activate, and commit an SMU package using a single command (1-step process) or using separate commands (3-step process).

Tip Use the 1-step process when you have to install just one SMU package file and use the 3-step process when you have to install multiple SMUs. The 3-step process minimises the number of reloads required when you have more than one SMU package file to install.

Installing an SMU Package: 1-Step Process

This task shows how to use the single **install add file activate commit** command for installing an SMU package.

Before you begin

Check that the SMU you are about to install corresponds to the software image installed on your device. For example, SMU `/auto/tftp-blr-users2/much/SMU_BUILDS/ie35xx-universalk9.2024-12 03_06.55_much.0.CSCwm26661.SSA.smu.bin` is compatible with software image `ie35xx-`**`17.17.1`**`.SPA.bin`.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**install add file flash:** *filename* [**activate commit**] **Example:** `Device# install add file flash:ie35xx-universalk9.2024-12 03_06.55_much.0.CSCwm26661.SSA.smu.bin activate commit`	Copies the maintenance update package from flash to the device, performs a compatibility check for the platform and image versions, activates the SMU package, and makes the package persistent across reloads. This command extracts the individual components of the .bin file into the subpackages and packages.conf files. You can also copy the SMU package from from a remote location (through FTP, HTTP, HTTPS, or TFTP). **Note** If the SMU file is copied using TFTP, use bootflash to activate the SMU.
Step 3	**exit** **Example:** `Device# exit`	Exits privileged EXEC mode and returns to user EXEC mode.

Installing an SMU Package: 3-Step Process

This task shows you the 3-step process for installing an SMU package. Use this method to install multiple SMUs and avoid multiple reloads.

Before you begin

Check that the SMU you are about to install corresponds to the software image installed on your device. For example, SMU `/auto/tftp-blr-users2/much/SMU_BUILDS/ie35xx-universalk9.2024-12 03_06.55_much.0.CSCwm26661.SSA.smu.bin` is compatible with software image `ie35xx-`**`17.17.1`**`.SPA.bin`.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**install add file** *location filename* **Example:** `Device# install add file flash:ie35xx-universalk9.2024-12 03_06.55_much.0.CSCwm26661.SSA.smu.bin Device# install add file flash:ie35xx-universalk9.2024-12 03_06.55_much.0.CSCwm26661.SSA.smu.bin`	Copies the maintenance update package from flash to the device, and then performs a compatibility check for the platform and image versions, and adds the SMU package on all member nodes or FRUs, as applicable. This command also runs base compatibility checks on a file to ensure that the SMU package is supported on the platform. It also adds an entry in the package/SMU.sta file, so that its status can be monitored and maintained. You can also copy the SMU package from a remote location (through FTP, HTTP, HTTPS, or TFTP).
Step 3	**install activate file** *location filename* **Example:** `Device# install activate file flash:ie35xx-universalk9.2024-12 03_06.55_much.0.CSCwm26661.SSA.smu.bin, ie35xx-universalk9.2024-12 03_06.55_much.0.CSCwm26661.SSA.smu.bin`	Activates the SMU package file that was added and updates the package status details. You will be promped to reload the system in order to complete the activation process. When entering multiple SMUs, use a comma (without a space before or after), to separate file names. Also ensure that total number of characters does not exceed 128. This step involves a reload.
Step 4	**install commit** **Example:** `Device# install commit`	Commits the activation changes to be persistent across reloads. The commit can be done after activation while the system is up, or after the first reload. If a package is activated but not committed, it remains active after the first reload, but not after the second reload.

Managing an SMU

This task shows how to rollback the installation state, deactivate, and remove a previously installed SMU package from the device. This can be used for a SMU that has been installed with the 1-step and 3-step process.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**install rollback to** {**base** | **committed** | **id** *commit-ID*} **Example:** `Device# install rollback to committed`	Returns the device to the previous installation state. After the rollback, a reload is required.
Step 3	**install deactivate file** *location filename* **Example:** `Device# install deactivate file flash:ie35xx-17.17.1.SPA.smu.bin`	Deactivates an active package, updates the package status, and triggers a process to restart or reload.
Step 4	**install remove** {**file** *location filename* | **inactive**} **Example:** `Device# install remove file flash:ie35xx-17.17.1.SPA.smu.bin`	Checks if the specified SMU is inactive and if it is, deletes it from the file system. The **inactive** option deletes all the inactive packages from the file system.
Step 5	**show version** **Example:** `Device# show version`	Displays the image version on the device.
Step 6	**show install summary** **Example:** `Device# show install summary`	Displays information about the active package. The output of this command varies according to the **install** commands that are configured.

Configuration Examples for Software Maintenance Upgrade

The following is a list of SMU configuration examples.

- Installing an SMU (3-Step Process, Using flash:), on page 2165
- Example: Installing an SMU (3-Step Process, Using TFTP), on page 2168

Installing an SMU (3-Step Process, Using flash:)

The following example shows how to install a SMU package by using the 3-step process. Here the SMU package file is saved in the device's flash.

1. Copying the SMU package file from flash and installing it.

```
Device# install add file flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
install_add: START Wed Mar 10 14:17:45 IST 2025
install_add: Adding SMU

--- Starting initial file syncing ---
Info: Finished copying flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin to the
selected switch(es)
Finished initial file syncing


*Mar 10 14:17:48.128 IST: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
 Started install add flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.binExecuting
pre scripts....
Executing pre sripts done.
--- Starting SMU Add operation ---
Performing SMU_ADD on all members
  [1] SMU_ADD package(s) on switch 1
  [1] Finished SMU_ADD on switch 1
Checking status of SMU_ADD on [1]
SMU_ADD: Passed on [1]
Finished SMU Add operation

SUCCESS: install_add /flash/ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin Wed Mar
10 14:18:00 IST 2025
```

Verifying the addition and installation of the SMU package file by using the **show install summary** command. The status of the SMU package file is `I`, because it has not been activated and committed yet.

```
Device# show install summary

[ Switch 1 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
IMG   C    17.17.1.0.3431

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

2. Activating the SMU package file.

```
Device# install activate file flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin

install_activate: START Wed Mar 10 14:19:59 IST 2025
install_activate: Activating SMU

*Mar 10 14:20:01.513 IST: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
 Started install activate flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin

This operation requires a reload of the system. Do you want to proceed? [y/n]y
Executing pre scripts....
Executing pre sripts done.

--- Starting SMU Activate operation ---
Performing SMU_ACTIVATE on all members
  [1] SMU_ACTIVATE package(s) on switch 1
  [1] Finished SMU_ACTIVATE on switch 1
```

```
Checking status of SMU_ACTIVATE on [1]
SMU_ACTIVATE: Passed on [1]
Finished SMU Activate operation

install_activate: Reloading the box to complete activation of the SMU...
install_activate will reload the system now!

*Mar 10 14:20:22.258 IST: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 1 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds
     Chassis 1 reloading, reason - Reload command
Mar 10 14:20:28.291: %PMAN-5-EXITACTION: F0/0: pvp: Process manager is exiting: reload
fp action requested
Mar 10 14:20:30.718: %PMAN-5-EXITACTION: R0/0: pvp: Proce
Mar 10 14:20:34.834: %PMAN-5-EXITACTION: C0/0: pvp: Process manager is exiting:
Mar 10 14:20:36.053: %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
 install activate SMU flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
watchdog watchdog0: watchdog did not stop!
reboot: Restarting system

Initializing Hardware...
<output truncated>

############
Mar 10 08:52:01.806: %BOOT-5-BOOTTIME_SMU_TEMP_ACTIVE_DETECTED: R0/0: install_engine:
SMU file /flash/ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin active temporary...
SMU committ is pending

Cisco IOS Software, L3 Switch Software (ie35xx_LITE_IOSXE), Version xx.x.x, RELEASE
SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2025 by Cisco Systems, Inc.
Compiled Thu 22-Mar-25 17:30 by mcpre

<output truncated>
```

Verifying activation of the SMU package file by using the **show install summary** command. The status of the SMU package file is U, because it has not been committed yet.

```
[ Switch 1 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   U   flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
IMG   C    17.16.4.0.3431

--------------------------------------------------------------------------------
Auto abort timer: active on install_activate, time before rollback - 01:41:52
--------------------------------------------------------------------------------
```

3. Committing the SMU package file

```
Device# install commit
install_commit: START Wed Mar 10 14:38:42 IST 2025
install_commit: Committing SMU

*Mar 10 14:38:44.906 IST: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
 Started install commitExecuting pre scripts....
Executing pre sripts done.
--- Starting SMU Commit operation ---
Performing SMU_COMMIT on all members
  [1] SMU_COMMIT package(s) on switch 1
  [1] Finished SMU_COMMIT on switch 1
Checking status of SMU_COMMIT on [1]
```

```
SMU_COMMIT: Passed on [1]
Finished SMU Commit operation

SUCCESS: install_commit /flash/ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin Wed Mar
 10 14:38:58 IST 2025
*Mar 10 14:38:59.385 IST: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 1 R0/0:
install_engine: Completed install commit SMU
```

Verifying the commit by using the **show install summary** command. The SMU package file has been installed, activated and committed and the status is C.

```
Device# show install summary
[ Switch 1 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C   flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
IMG   C    17.17.1.0.3431

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

Verifying active packages by using the **show install active** command

```
Device# show install active
[ Switch 1 ] Active Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C    flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
IMG   C    17.17.1.0.3431
```

Checking the version, by using the **show version** command:

```
Device# show version
Cisco IOS XE Software, Version 17.17.1
Cisco IOS Software, L3 Switch Software (ie35xx_LITE_IOSXE), Version 17.17.1, RELEASE
SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2025 by Cisco Systems, Inc.
Compiled Thu 22-Mar-25 17:30 by mcpre
...
```

Example: Installing an SMU (3-Step Process, Using TFTP)

The following example shows how to install a SMU package by using the 3-step process. Here the SMU package file is saved in a remote (TFTP) location.

1. Adding the SMU package file.

```
Device# install add file
tftp://172.16.0.1//tftpboot/folder1/ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin

Mar 22 11:32:27.035: %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 add tftp://172.16.0.1//tftpboot/folder1/ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
```

```
Mar 22 11:32:27.035 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 add tftp://172.16.0.1//tftpboot/folder1/ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
Downloading file
tftp://172.16.0.1//tftpboot/folder1/ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
Finished downloading file
tftp://172.16.0.1//tftpboot/folder1/ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
to flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
install_add: Adding SMU
install_add: Checking whether new add is allowed ....

--- Starting initial file syncing ---

025335: *Mar 22 2025 11:32:26 UTC: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0:
install_engine: Started install add
tftp://172.16.0.1//tftpboot/folder1/ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin[1]:
 Copying flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin from switch 1 to switch
 2
[2]: Finished copying to switch 2
Info: Finished copying flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin to the
selected switch(es)
Finished initial file syncing

--- Starting SMU Add operation ---
Performing SMU_ADD on all members
[1] SMU_ADD package(s) on switch 1
[1] Finished SMU_ADD on switch 1
[2] SMU_ADD package(s) on switch 2
[2] Finished SMU_ADD on switch 2
Checking status of SMU_ADD on [1 2]
SMU_ADD: Passed on [1 2]
Finished SMU Add operation

SUCCESS: install_add Mon Mar 22 11:32:56 UTC 2025
Mar 22 11:32:57.598: %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
 install add SMU flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
Mar 22 11:32:57.598 %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
install add SMU flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin

ECSG-SEC-35xx-24P#
025336: *Mar 22 2025 11:32:57 UTC: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 1 R0/0:
install_engine: Completed install add SMU
flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
```

Verifying addition by using the **show install summary** command.

```
Device# show install summary
[ Switch 1 2 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type St Filename/Version
--------------------------------------------------------------------------------
SMU I flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
IMG C 17.17.02.0.6

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

2. Activating the SMU package file.

Note You use TFTP to add the SMU package file (in the previous step) and *flash*, to activate - not TFTP.

```
Device# install activate file flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin

install_activate: START Mon Mar 22 11:37:17 UTC 2025

Mar 22 11:37:37.582: %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 activate flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
Mar 22 11:37:37.582 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 activate flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
install_activate: Activating SMU

025337: *Mar 22 2025 11:37:37 UTC: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0:
install_engine: Started install activate
flash:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
This operation may require a reload of the system. Do you want to proceed? [y/n]n
```

Checking the version, by using the **show version** command:

```
Device# show version
Cisco IOS XE Software, Version 17.17.1
Cisco IOS Software, L3 Switch Software (ie35xx_LITE_IOSXE), Version 17.17.1, RELEASE
SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2025 by Cisco Systems, Inc.
Compiled Thu 22-Mar-25 17:30 by mcpre
<output truncated>
```

3. Committing the SMU package file.

```
Device# install commit

install_commit: START Mon Mar 22 11:38:48 UTC 2025
SUCCESS: install_commit Mon Mar 22 11:38:52 UTC 2025
Device#
```

Verifying that the update package is now committed, and that it will be persistent across reloads:

```
Device# show install summary

Active Packages:
tftp:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
Inactive Packages:
No packages
Committed Packages:
tftp:ie35xx_lite_iosxe.xx.xx.xx.CSCvk70181.SPA.smu.bin
Uncommitted Packages:
No packages
Device#
```

CHAPTER 145

Working with the Flash File System

Information About the Flash File System

The flash file system is a single flash device on which you can store files. It also provides several commands to help you manage software bundles and configuration files. The default flash file system on the device is named flash:.

As viewed from the active device, flash: refers to the local flash device, which is the device attached to the same device on which the file system is being viewed.

Only one user at a time can manage the software bundles and configuration files.

Displaying Available File Systems

To display the available file systems on your device, use the **show file systems** privileged EXEC command as shown in this example for a standalone device:

```
Device# show file systems
File Systems:
Size(b) Free(b) Type Flags Prefixes
- - opaque rw system:
- - opaque rw tmpsys:
1651314688 1467920384 disk rw crashinfo:
* 11353194496 6942072832 disk rw flash:
7723847680 7646384128 disk ro webui:
- - opaque rw null:
- - opaque ro tar:
- - network rw tftp:
```

```
2097152 2089932 nvram rw nvram:
- - network rw rcp:
- - network rw http:
- - network rw ftp:
- - network rw scp:
- - network rw https:
- - opaque ro cns:
118014062592 111933124608 disk rw usbflash1:
```

This example displays the usbflash1 filesystem format.

```
Device#show usbflash1: filesys
Filesystem: usbflash1
Filesystem Path: /vol/usb1
Filesystem Type: ext4
Mounted: Read/Write
```

Table 147: show file systems Field Descriptions

Field	Value
Size(b)	Amount of memory in the file system in bytes.
Free(b)	Amount of free memory in the file system in bytes.
Type	Type of file system. **disk**—The file system is for a flash memory device, USB flash, and crashinfo file. **network**—The file system for network devices; for example, an FTP server or and HTTP server. **nvram**—The file system is for a NVRAM device. **opaque**—The file system is a locally generated pseudo file system (for example, the system) or a download interface, such as brimux. **unknown**—The file system is an unknown type.
Flags	Permission for file system. **ro**—read-only. **rw**—read/write. **wo**—write-only.

Field	Value
Prefixes	Alias for file system. **crashinfo:**—Crashinfo file. **flash:**—Flash file system. **ftp:**—FTP server. **http:**—HTTP server. **https:**—Secure HTTP server. **nvram:**—NVRAM. **null:**—Null destination for copies. You can copy a remote file to null to find its size. **rcp:**—Remote Copy Protocol (RCP) server. **scp:**—Session Control Protocol (SCP) server. **system:**—Contains the system memory, including the running configuration. **tftp:**—TFTP network server. **usbflash0:**—USB flash memory. **usbflash1:**—External USB flash memory. **ymodem:**—Obtain the file from a network machine by using the Ymodem protocol.

Setting the Default File System

You can specify the file system or directory that the system uses as the default file system by using the **cd** *filesystem:* privileged EXEC command. You can set the default file system to omit the *filesystem:* argument from related commands. For example, for all privileged EXEC commands that have the optional *filesystem:* argument, the system uses the file system specified by the **cd** command.

By default, the default file system is *flash:*.

You can display the current default file system as specified by the **cd** command by using the **pwd** privileged EXEC command.

Displaying Information About Files on a File System

You can view a list of the contents of a file system before manipulating its contents. For example, before copying a new configuration file to flash memory, you might want to verify that the file system does not already contain a configuration file with the same name. Similarly, before copying a flash configuration file to another location, you might want to verify its filename for use in another command. To display information about files on a file system, use one of the privileged EXEC commands listed in the following table.

Table 148: Commands for Displaying Information About Files

Command	Description
dir [/all] **[***filesystem:filename***]**	Displays a list of files on a file system.
show file systems	Displays more information about each of the files on a file system.
show file information *file-url*	Displays information about a specific file.
show file descriptors	Displays a list of open file descriptors. File descriptors are the internal representations of open files. You can use this command to see if another user has a file open.

For example, to display a list of all files in a file system, use the **dir** privileged EXEC command:

```
Device# dir flash:
DDirectory of bootflash:/

616513  drwx             4096  Jul 15 2015 07:11:35 +00:00  .installer
608402  -rw-            33818  Sep 25 2015 11:41:35 +00:00  bootloader_evt_handle.log
608403  drwx             4096  Feb 27 2017 13:56:47 +00:00  .ssh
608410  -rw-                0   Jun 5 2015 10:16:17 +00:00  dc_stats.txt
608411  drwx            20480  Sep 23 2015 11:50:13 +00:00  core
624625  drwx             4096  Sep 23 2015 12:29:27 +00:00  .prst_sync
640849  drwx             4096  Feb 27 2017 13:57:30 +00:00  .rollback_timer
608412  drwx             4096  Jun 17 2015 18:12:47 +00:00  orch_test_logs
608413  -rw-         33554432  Sep 25 2015 11:43:15 +00:00  nvram_config
608417  -rw-               35  Sep 25 2015 20:17:42 +00:00  pnp-tech-time
608439  -rw-           214054  Sep 25 2015 20:17:48 +00:00  pnp-tech-discovery-summary
608419  drwx             4096  Jul 23 2015 07:50:25 +00:00  util
616514  drwx             4096  Mar 18 2015 11:09:04 +00:00  onep
608442  -rw-              556  Mar 18 2015 11:19:34 +00:00  vlan.dat
608448  -rw-          1131779  Mar 28 2015 13:13:48 +00:00  log.txt
616516  drwx             4096   Apr 1 2015 09:34:56 +00:00  gs_script
616517  drwx             4096   Apr 6 2015 09:42:38 +00:00  tools
608440  -rw-              252  Sep 25 2015 11:41:52 +00:00  boothelper.log
624626  drwx             4096  Apr 17 2015 06:10:55 +00:00  SD_AVC_AUTO_CONFIG
608488  -rw-            98869  Sep 25 2015 11:42:15 +00:00  memleak.tcl
608437  -rwx            17866  Jul 16 2015 04:01:10 +00:00  ardbeg_x86
632745  drwx             4096  Aug 20 2015 11:35:09 +00:00  CRDU
632746  drwx             4096  Sep 16 2015 08:57:44 +00:00  ardmore
608418  -rw-          1595361   Jul 8 2015 11:18:33 +00:00
system-report_RP_0_20150708-111832-UTC.tar.gz
608491  -rw-         67587176  Aug 12 2015 05:30:35 +00:00  mcln_x86_kernel_20170628.SSA
608492  -rwx         74880100  Aug 12 2015 05:30:57 +00:00  stardust.x86.idprom.0718B

11250098176 bytes total (9128050688 bytes free)
Device#
```

Changing Directories and Displaying the Working Directory

Follow these steps to change directories and to display the working directory:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**dir** *filesystem:* **Example:** `Device# dir flash:`	Displays the directories on the specified file system. For *filesystem:*, use flash: for the system board flash device.
Step 3	**cd** *directory_name* **Example:** `Device# cd new_configs`	Navigates to the specified directory. The command example shows how to navigate to the directory named *new_configs*.
Step 4	**pwd** **Example:** `Device# pwd`	Displays the working directory.
Step 5	**cd** **Example:** `Device# cd`	Navigates to the default directory.

Creating Directories

Beginning in privileged EXEC mode, follow these steps to create a directory:

Procedure

	Command or Action	Purpose
Step 1	**dir** *filesystem:* **Example:** `Device# dir flash:`	Displays the directories on the specified file system. For *filesystem:*, use flash: for the system board flash device.
Step 2	**mkdir** *directory_name* **Example:**	Creates a new directory. Directory names are case sensitive and are limited to 45 characters between the slashes (/); the name cannot contain

	Command or Action	Purpose
	`Device# mkdir new_configs`	control characters, spaces, slashes, quotes, semicolons, or colons.
Step 3	**dir** *filesystem:* **Example:** `Device# dir flash:`	Verifies your entry.

Removing Directories

To remove a directory with all its files and subdirectories, use the **delete /force /recursive** *filesystem:/file-url* privileged EXEC command.

Use the **/recursive** keyword to delete the named directory and all subdirectories and the files contained in it. Use the **/force** keyword to suppress the prompting that confirms a deletion of each file in the directory. You are prompted only once at the beginning of this deletion process.

For *filesystem*, use **flash:** for the system board flash device. For *file-url*, enter the name of the directory to be deleted. All of the files in the directory and the directory are removed.

Caution When directories are deleted, their contents cannot be recovered.

Copying Files

To copy a file from a source to a destination, use the **copy** *source-url destination-url* privileged EXEC command. For the source and destination URLs, you can use **running-config** and **startup-config** keyword shortcuts. For example, the **copy running-config startup-config** command saves the currently running configuration file to the NVRAM section of flash memory to be used as the configuration during system initialization.

You can also copy from special file systems (**xmodem:**, **ymodem:**) as the source for the file from a network machine that uses the Xmodem or Ymodem protocol. SSH File Transfer Protocol (SFTP) is also another option to copy switch configuration or image files. For more information, refer the *Configuring SSH File Transfer Protocol* chapter of the *Security Configuration Guide*.

Network file system URLs include ftp:, rcp:, tftp:, scp:, http:, and https: and have these syntaxes:

- FTP—ftp:[[//username [:password]@location]/directory]/filename
- RCP—rcp:[[//username@location]/directory]/filename
- TFTP—tftp:[[//location]/directory]/filename
- SCP—scp:[[//username [:password]@location]/directory]/filename
- HTTP—http:[[//username [:password]@location]/directory]/filename
- HTTPS—https:[[//username [:password]@location]/directory]/filename

Note The password must not contain the special character '@'. If the character '@' is used, the copy fails to parse the IP address of the server.

Local writable file systems include flash:.

Some invalid combinations of source and destination exist. Specifically, you cannot copy these combinations:

- From a running configuration to a running configuration
- From a startup configuration to a startup configuration

Deleting Files

When you no longer need a file on a flash memory device, you can permanently delete it. To delete a file or directory from a specified flash device, use the **delete** [**/force**] [**/recursive**] [*filesystem:*]*/file-url* privileged EXEC command.

Use the **/recursive** keyword for deleting a directory and all subdirectories and the files contained in it. Use the **/force** keyword to suppress the prompting that confirms a deletion of each file in the directory. You are prompted only once at the beginning of this deletion process. Use the **/force** and **/recursive** keywords for deleting old software images that were installed by using the **archive download-sw** command but are no longer needed.

If you omit the *filesystem:* option, the device uses the default device specified by the **cd** command. For *file-url*, you specify the path (directory) and the name of the file to be deleted.

When you attempt to delete any files, the system prompts you to confirm the deletion.

Caution When files are deleted, their contents cannot be recovered.

This example shows how to delete the file *myconfig* from the default flash memory device:

```
Device# delete myconfig
```

Creating, Displaying and Extracting Files

You can create a file and write files into it, list the files in a file, and extract the files from a file as described in the next sections.

Beginning in privileged EXEC mode, follow these steps to create a file, display the contents, and extract it:

Procedure

	Command or Action	Purpose
Step 1	**archive tar /create** *destination-url* **flash:** */file-url*	Creates a file and adds files to it.

	Command or Action	Purpose
	Example: `Device# archive tar /create tftp:172.20.10.30/saved. flash:/new-configs`	For destination-url, specify the destination URL alias for the local or network file system and the name of the file to create: • Local flash file system syntax: **flash:** • FTP syntax: **ftp**:[[//*username*[:*password*]@*location*]/*directory*]/*-filename*. • RCP syntax: **rcp**:[[//*username@location*]/*directory*]/*-filename*. • TFTP syntax: **tftp**:[[//*location*]/*directory*]/*-filename*. For **flash:***/file-url*, specify the location on the local flash file system in which the new file is created. You can also specify an optional list of files or directories within the source directory to add to the new file. If none are specified, all files and directories at this level are written to the newly created file.
Step 2	**archive tar /table** *source-url* **Example:** `Device# archive tar /table flash: /new_configs`	Displays the contents of a file. For *source-url*, specify the source URL alias for the local or network file system. The *-filename.* is the file to display. These options are supported: • Local flash file system syntax: **flash:** • FTP syntax: **ftp**:[[//*username*[:*password*]@*location*]/*directory*]/*-filename*. • RCP syntax: **rcp**:[[//*username@location*]/*directory*]/*-filename*. • TFTP syntax: **tftp**:[[//*location*]/*directory*]/*-filename*. You can also limit the file displays by specifying a list of files or directories after the file. Only those files appear. If none are specified, all files and directories appear.
Step 3	**archive tar /xtract** *source-url* **flash:***/file-url* [*dir/file...*] **Example:** `Device# archive tar /xtract`	Extracts a file into a directory on the flash file system. For *source-url*, specify the source URL alias for the local file system. The *-filename.* is the

	Command or Action	Purpose
	`tftp:/172.20.10.30/saved.` `flash:/new-configs`	file from which to extract files. These options are supported: • Local flash file system syntax: **flash:** • FTP syntax: **ftp**:[[//*username*[:*password*][@*location*]/*directory*]/*-filename*. • RCP syntax: **rcp**:[[//*username@location*]/*directory*]/*-filename*. • TFTP syntax: **tftp**:[[//*location*]/*directory*]/*-filename*. For **flash:**/*file-url* [*dir/file...*], specify the location on the local flash file system from which the file is extracted. Use the *dir/file...* option to specify a list of files or directories within the file to be extracted. If none are specified, all files and directories are extracted.
Step 4	**more** [/**ascii** | /**binary** | /**ebcdic**] /*file-url* **Example:** `Device# more` `flash:/new-configs`	Displays the contents of any readable file, including a file on a remote file system.

CHAPTER 146

Performing Factory Reset

Prerequisites for Performing a Factory Reset

- Ensure that all the software images, including the current image, configurations, and personal data are backed up before you begin the factory reset process.
- Ensure that there is uninterrupted power supply when the factory reset process is in progress.
- Ensure that In-Service Software Upgrade (ISSU) or In-Service Software Downgrade (ISSD) are not in progress before you begin the factory reset process.

Restrictions for Performing a Factory Reset

- Software patches, if installed on the device, will not be restored after the factory reset process.
- If the **factory-reset** command is issued through a VTY session, the session is not restored after completion of the factory reset process.

Information About Performing a Factory Reset

Factory reset erases all the customer-specific data stored in a device and restores the device to its original configuration at the time of shipping. Data that is erased includes configurations, log files, boot variables, core files, and credentials such as Federal Information Processing Standard-related (FIPS-related) keys. The erasure is consistent with the clear method, as described in NIST SP 800-88 Rev. 1.

The factory reset process is used in the following scenarios:

- Return Material Authorization (RMA) for a device: If you have to return a device to Cisco for RMA, remove all the customer-specific data before obtaining an RMA certificate for the device.

- Recovering a compromised device: If the key material or credentials that are stored on a device are compromised, reset the device to the factory configuration, and then reconfigure the device.

During a factory reset, the device reloads and enters ROMMON mode. After the factory reset, the device removes all its environment variables, including the **MAC_ADDRESS** and the **SERIAL_NUMBER** variables, which are required to locate and load the software. Perform a reset in ROMmon mode to automatically set the environment variables. The BAUD rate environment variable returns to its default value after a factory reset. Make sure that the BAUD rate and the console speed are the same at all times. Otherwise, the console becomes unresponsive.

After the system reset in ROMmon mode is complete, add the Cisco IOS image either through an USB or TFTP.

The following table provides details about the data that is erased and retained during the factory reset process:

Table 149: Data Erased and Retained During Factory Reset

Data Erased	Data Retained
All Cisco IOS images, including the current boot image	Data from remote field-replaceable units (FRUs)
Crash information and logs	Value of the configuration register.
User data, startup and running configuration, and contents of removable storage devices, such as Serial Advanced Technology Attachment (SATA), Solid State Drive (SSD), or USB	—
Credentials such as FIPS-related keys	Credentials such as Secure Unique Device Identifier (SUDI) certificates, and public key infrastructure (PKI) keys.
Onboard Failure Logging (OBFL) logs	
ROMmon variables added by a user.	—

Secure Data Wipe

The device storage is used to maintain software images, device configuration, software logs and operational history. Customer-specific data can be contained in any of these areas. The information can include network architecture and design used by customers.

The **all secure** option in the **factory-reset** command performs data sanitization and securely resets the device. After data sanitization, the device reloads and boots with the software image present in flash.

Secure data wipe feature implements guidelines for media sanitization as described in NIST SP 800-88 Rev. 1.

How to Perform a Factory Reset

To perform a factory reset, complete this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	• For a standalone device: **factory-reset** {**all** [**secure**] [**3-pass**] | **config** | **boot-vars**} **Example:** `Device# factory-reset all`	Resets the device to its configuration at the time of its shipping. No system configuration is required to use the **factory reset** command. The following options are available: • **all**: Erases all the content from the NVRAM, all the Cisco IOS images, including the current boot image, boot variables, startup and running configuration data, and user data. We recommend that you use this option. • **all secure**: Performs data sanitization and securely resets the device. **Note** • You can use the **all secure** option only on standalone devices. • This option implements guidelines for media sanitization as described in NIST SP 800-88 Rev. 1. • The **factory-reset all secure** command initiates data sanitization. The booted image of the device is retained. • When data sanitization is completed, the device reloads, and the device image is retained in flash if it was booted with an image from the flash. • **secure 3-pass**: Erases all the content from the device with 3-pass overwrite. • Pass 1: Overwrites all addressable locations with binary zeroes. • Pass 2: Overwrites all addressable locations with binary ones.

	Command or Action	Purpose
		• Pass 3: Overwrites all addressable locations with a random bit pattern. **Note** This option takes approximately thrice the time taken to perform any other option. • **config**: Resets the startup configurations. • **boot-vars**: Resets the user-added boot variables. After the factory reset process is successfully completed, the device reboots and enters ROMmon mode.

Configuration Examples for Performing a Factory Reset

The following example shows how to perform a factory reset on a standalone switch:

```
Device> enable
Device# factory-reset all

The factory reset operation is irreversible for all operations. Are you sure? [confirm]
The following will be deleted as a part of factory reset:
1: Crash info and logs
2: User data, startup and running configuration
3: All IOS images, including the current boot image
4: OBFL logs
5: User added rommon variables
6: Data on Field Replaceable Units(USB/SSD/SATA)
The system will reload to perform factory reset.
It will take some time to complete and bring it to rommon.
You will need to load IOS image using USB/TFTP from rommon after
this operation is completed.
DO NOT UNPLUG THE POWER OR INTERRUPT THE OPERATION
Are you sure you want to continue? [confirm]
```

The following sample output from the **show platform software factory-reset secure log** command displays the data sanitization report:

```
Device# show platform software factory-reset secure log
Factory reset log:
#CISCO IE35xx DATA SANITIZATION REPORT#
START : 18-09-2022, 06:18:44
  END : 18-09-2022, 06:23:36
-MTD-
PNM : nor
NIST : PURGE
-eMMC-
MID : 'Micron'
PNM : 'Q2J55L'
```

```
SN : 0x00000001
NIST : PURGE
```

CHAPTER 147

Configuring Secure Storage

Information About Secure Storage

Secure Storage feature allows you to secure critical configuration information by encrypting it. It encrypts asymmetric key-pairs, pre-shared secrets, the type 6 password encryption key and certain credentials. An instance-unique encryption key is stored in the hardware trust anchor to prevent it from being compromised.

Enabling Secure Storage

Before you begin

By default, this feature is disabled.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters the global configuration mode.
Step 2	**service private-config-encryption** **Example:** `DEvice(config)# service private-config-encryption`	Enables the Secure Storage feature on your device.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 4	**write memory** **Example:** `Device# write memory`	Encrypts the private-config file and saves the file in an encrypted format.

Disabling Secure Storage

Before you begin

To disable Secure Storage feature on a device, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters the global configuration mode.
Step 2	**no service private-config-encryption** **Example:** `Device(config)# no service private-config-encryption`	Disables the Secure Storage feature on your device. When secure storage is disabled, all the user data is stored in plain text in the NVRAM.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 4	**write memory** **Example:** `Device# write memory`	Decrypts the private-config file and saves the file in plane format.

Verifying the Status of Encryption

Use the **show parser encrypt file status** command to verify the status of encryption. The following command output indicates that the feature is available but the file is not encrypted. The file is in 'plain text' format.

This is an example of unencrypted file and shows the status here as plain text.

```
Device#show parser encrypt file status
Feature: Enabled
File Format: Plain Text
Encryption Version: Ver1
```

CHAPTER 148

Trace Management

Information About Trace Management

The tracing functionality logs internal events. Trace files are automatically created and saved on the persistent storage device of specific platforms.

If the device has issues, the contents of the trace files are useful to troubleshoot the issue. The trace file outputs provide logs that are used to locate and solve the issue, and helps to get a detailed view of system actions and operations.

To view the recent trace information for a specific process, use the **show logging [process | Profile | process-helper]** command. The **process** keyword uses the first few letters of the name of a process and provides trace logs of the process that starts or matches with the entered string, the **profile** keyword lists the predefined set of process names, and the **profile-helper** keyword displays the available names.

To change the verbosity in a trace message output, you can adjust the trace level of processes using the **set platform software trace level** command. You can choose the **all** keyword to adjust the trace level for all the processes listed or you can select a specific process. When you select a specific process, there's also the option to adjust the trace level for a specific module, or you can use the **all-modules** keyword to adjust all the modules of processes.

Introduction to Binary Tracing

Binary tracing is helpful in gathering trace information with a minimal impact on performance. In binary tracing, the tracing is always on for the system components and a basic level of trace is collected on all the time; thus, the data necessary for troubleshooting a problem has been captured the first time it occurs.

Introduction to Conditional Debugging and Radioactive Tracing

The Conditional Debugging feature allows you to enable debugging and logging for specific features based on the set of conditions you define. This feature is useful in systems where many features are supported.

The Conditional debug allows granular debugging in a network that is operating at a large scale with a large number of features. It allows you to observe detailed debugs for granular instances within the system. This

type of debugging is useful when we need to debug only a particular session among thousands of sessions. It's also possible to specify multiple conditions.

A condition refers to a feature or identity, where an identity could be an interface, IP Address, or a MAC address and so on.

Conditional debugging is in contrast to the general debug command, that produces its output without discriminating on the feature objects that are being processed. General debug command consumes numerous system resources and impacts the system performance.

Radioactive tracing provides the ability to form a chain of execution for operations of interest across the system, at an increased verbosity level. This provides a way to print conditionally debug information (up to DEBUG Level or a specified level) across threads, processes, and function calls.

Radioactive Tracing when coupled with Conditional Debugging, provides a single debug command to debug all execution contexts related to the condition. You can execute this command without being aware of the various control flow processes of the feature within the box and without having to issue debugs at these processes individually.

Tracing Levels

Trace level determines the types of traces outputted. Each trace message is assigned a trace level. If the trace level of a process or its module it set as greater than or equal to the level as the trace message, the trace message is displayed otherwise, it's skipped. For example, the default trace level is **Notice** level, so all traces with the **Notice** level and below the notice level are included while the traces above the **Notice** level are excluded.

The following table shows the available tracing levels, and provides descriptions of the message that are displayed with each tracing level. The tracing levels listed in the table are from the lowest to the highest order. The default trace level is **Notice**.

Table 150: Tracing Levels and Descriptions

Tracing Level	Description
Fatal	The message stating the process is aborted.
Emergency	The message is regarding an issue that makes the system unusable.
Alert	The message indicating that an action must be taken immediately.
Critical	The message is regarding a critical event causing loss of important functions.
Error	The message is regarding a system error.
Warning	The message is regarding a system warning.
Notice	The message is regarding a significant event.
Informational	The message is useful for informational purposes only.
Debug	The message provides debug-level output.

Tracing Level	Description
Verbose	All possible trace messages are sent.
Noise	All possible trace messages for the module are logged. The noise level is always equal to the highest possible tracing level. Even if a future enhancement to tracing introduces a higher tracing level, the noise level will become equal to the level of that new enhancement.

Payload Filter

This feature is used to filter trace messages. Trace messages contain actual debug information such as text strings, special characters, and variable arguments (strings), integers, long, IPv4/IPv6/MAC addresses, and so on. Using the payload feature, the trace messages can be filtered based on the selected criteria and without string operations.

You can use the following set and show commands to configure a payload filter and to view the applied filters.

Table 151: Set Commands for Payload Filter

set platform software btrace-manager ... utm-pf enable	Enables and disables the payload filtering feature.
set platform software btrace-manager ... utm-pf disable	
set platform software btrace-manager ... consumer-name <input> create	Creates and deletes consumer/stream.
set platform software btrace-manager ... consumer-name <input> delete	
set platform software btrace-manager ... consumer-name <input> filter <input> add	Applies and removes filter on stream/consumer
set platform software btrace-manager ... consumer-name <input> filter <input> remove	

Table 152: Show Commands for Payload Filter

#show platform software btrace-manager ... utm-pf	Shows the current status of the payload feature and other additional details
show platform software btrace-manager ... utm-pf consumer-name <input> all-filters	Shows all filters currently applied on consumer/stream.
show platform software btrace-manager ... utm-pf consumer-name <input> all-luids	Shows all or selected LUID of consumer for the applied filter.
show platform software btrace-manager ... utm-pf consumer-name <input> filter <input>	

show platform software btrace-manager ... utm-pf message	Shows consumer/stream messages.

How to Configure Conditional Debugging

Conditional Debugging and Radioactive Tracing

Radioactive Tracing when coupled with Conditional Debugging, prpvides a single debug command to debug all execution contexts related to the condition. You can execute this command without being aware of the various control flow processes of the feature within the box and without having to issue debugs at these processes individually.

Configuring Conditional Debugging

Follow the steps to configure conditional debugging:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug platform condition mac** {*mac-address*} **Example:** `Device# debug platform condition mac bc16.6509.3314`	Configures conditional debugging for the MAC Address specified.
Step 3	**debug platform condition start** **Example:** `Device# debug platform condition start`	Starts conditional debugging (this step starts radioactive tracing if there's a match on one of the preceding conditions).
Step 4	**show platform condition** OR **show debug** **Example:** `Device# show platform condition` `Device# show debug`	Displays the current conditions set.
Step 5	**debug platform condition stop** **Example:** `Device# debug platform condition stop`	Stops conditional debugging (this step stops radioactive tracing).

	Command or Action	Purpose
Step 6	**request platform software trace archive** [**last** {*number*} **days**] [**target** {*crashinfo:* | *flashinfo:*}] **Example:** `# request platform software trace archive last 2 days`	(Optional) Displays historical logs of merged tracefiles on the system. Filter on any combination of number of days or location.
Step 7	**show platform software trace** [**filter-binary** | **level** | **message**] **Example:** `Device# show platform software trace message`	(Optional) Displays logs merged from the latest trace file. Filter on any combination of application condition, trace module name, and trace level. • **filter-binary** - Filter the modules to be collated • **level** - Show trace levels • **message** - Show trace message ring contents **Note** On the device: • Available from IOS console in addition to linux shell. • Generates a file with merged logs • Displays merged logs only from staging area.
Step 8	**clear platform condition all** **Example:** `Device# clear platform condition all`	Clears all conditions.

What to do next

Note The commands **request platform software trace filter-binary** and **show platform software trace filter-binary** work in a similar way. The only difference is:

- **request platform software trace filter-binary** - Sources the data from historical logs.
- **show platform software trace filter-binary** – Sources the data from the flash Temp directory.

The *mac_log <..date..>* is the important file, as it provides messages for the MAC that is being debugged. The command **show platform software trace filter-binary** also generates the same flash files, and also prints the mac_log on the screen.

Collecting Trace Files

To collect trace files from a device, follow these steps:

1. To request the tracelogs for a specific time period (For example: Five days), use the command:

   Device# **request platform software trace archive last 5 day**

2. The system generates a tar ball (.gz file) of the tracelogs in the location **/flash**:

Copying Archived Trace Files

The following is an example of the trace file for a switching device:

```
Device# dir crashinfo:/tracelogs
Directory of crashinfo:/tracelogs/

50664 -rwx 760 Sep 22 2015 11:12:21 +00:00 plogd_F0-0.bin_0.gz
50603 -rwx 991 Sep 22 2015 11:12:08 +00:00 fed_pmanlog_F0-0.bin_0.9558.20150922111208.gz
50610 -rw- 11 Nov 2 2015 00:15:59 +00:00 timestamp
50611 -rwx 1443 Sep 22 2015 11:11:31 +00:00
auto_upgrade_client_sh_pmanlog_R0-.bin_0.3817.20150922111130.gz
50669 -rwx 589 Sep 30 2015 03:59:04 +00:00 cfgwr-8021_R0-0.bin_0.gz
50612 -rwx 1136 Sep 22 2015 11:11:46 +00:00 reflector_803_R0-0.bin_0.1312.20150922111116.gz
50794 -rwx 4239 Nov 2 2015 00:04:32 +00:00 IOSRP_R0-0.bin_0.14239.20151101234827.gz
50615 -rwx 131072 Nov 2 2015 00:19:59 +00:00 linux_iosd_image_pmanlog_R0-0.bin_0
--More—
```

You can copy the trace files using one of the following options:

```
Device# copy crashinfo:/tracelogs ?
  crashinfo: Copy to crashinfo: file system
  flash: Copy to flash: file system
  ftp: Copy to ftp: file system
  http: Copy to http: file system
  https: Copy to https: file system
  null: Copy to null: file system
  nvram: Copy to nvram: file system
  rcp: Copy to rcp: file system
  running-config Update (merge with) current system configuration
  scp: Copy to scp: file system
  startup-config Copy to startup configuration
  syslog: Copy to syslog: file system
  system: Copy to system: file system
  tftp: Copy to tftp: file system
  tmpsys: Copy to tmpsys: file system
```

The general syntax for copying onto a TFTP server is as follows:

```
Device# copy source: tftp:
Device# copy crashinfo:/tracelogs/IOSRP_R0-0.bin_0.14239.20151101234827.gz tftp:
Address or name of remote host []? 2.2.2.2
Destination filename [IOSRP_R0-0.bin_0.14239.20151101234827.gz]?
```

Note It's important to clear the generated report or archive files off the device so that there's flash space available for tracelog and other purposes.

Configuring Payload Filter

To configure a payload filter, you must create a consumer and add the relevant payload filter data.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**set platform software btrace-manager utm-pf enable** **Example:** `Device# set platform software btrace-manager chassis active r0 utm-pf enable` `Device# set platform software btrace-manager chassis active r0 utm-pf disable`	Enables or disables the payload filter.
Step 3	**set platform software btrace-manager** {*consumer-name*} **create** **Example:** `Device# set platform software btrace-manager chassis active r0 consumer-name utm_pf_test create`	Creates a consumer name.
Step 4	**set platform software btrace-manager consumer** {*consumer-name*} **filter** {*input*} **add** **Example:** `Device# set platform software btrace-manager chassis active r0 consumer-name utm_pf_test filter "Failed to retrieve an interface" add`	Add a filter data.

Configuration Examples for Trace Management

The following is an output example of the *show platform condition* command.

The following is a sample of the *debug platform condition stop* command.

```
Device# debug platform condition stop
Conditional Debug Global State: Stop
```

The following is an example of the **show logging** command for the **ios** process.

```
Device# show logging process ios
Logging display requested on 2022/10/27 09:32:06 (PDT) for Hostname: [vwlc_1_9222], Model:
 [C9800-CL-K9], Version: [X.X.X], SN: [9ZY0U03YBM0], MD_SN: [9ZY0U03YBM0]

Displaying logs from the last 0 days, 0 hours, 10 minutes, 0 seconds
executing cmd on chassis 1 ...
Unified Decoder Library Init .. DONE
Found 1 UTF Streams

2022/10/27 09:31:52.835197577 {iosrp_R0-0}{1}: [parser_cmd] [26471]: (note): id=
console@console:user= cmd: 'show logging process ios' SUCCESS 2022/10/27 08:31:48.762 PST
2022/10/27 09:31:59.651965736 {iosrp_R0-0}{1}: [parser_cmd] [26471]: (note): id=
console@console:user= cmd: 'show logging process ios internal' SUCCESS 2022/10/27 08:31:56.485
 PST
============================================================
======= Unified Trace Decoder Information/Statistics ======
============================================================
----------------- Decoder Input Information --------------
============================================================
 Num of Unique Streams .. 1
 Total UTF To Process ... 1
 Total UTM To Process ... 75403
 UTM Process Filter ..... ios
 MRST Filter Rules ...... 4
============================================================
----------------- Decoder Output Information -------------
============================================================
 First UTM TimeStamp ............... 2022/10/27 02:21:47.048461994
 Last UTM TimeStamp ................ 2022/10/27 09:32:04.919540850
 UTM [Skipped / Rendered / Total] .. 75401 / 2 / 75403
 UTM [ENCODED] ..................... 75266
 UTM [PLAIN TEXT] .................. 94
 UTM [DYN LIB] ..................... 0
 UTM [MODULE ID] ................... 0
 UTM [TDL TAN] ..................... 43
 UTM [APP CONTEXT] ................. 0
 UTM [MARKER] ...................... 0
 UTM [PCAP] ........................ 0
 UTM [LUID NOT FOUND] .............. 0
============================================================
```

The following is an example of the **show logging profile wireless** command.

```
Device# show logging profile wireless
Logging display requested on 2023/03/13 09:07:09 (UTC) for Hostname: [FABRIEK], Model:
[C8300-1N1S-4T2X], Version: [X.X.X], SN: [FDO24190V85], MD_SN: [FDO2451M13G]

Displaying logs from the last 0 days, 0 hours, 10 minutes, 0 seconds
executing cmd on chassis local ...
Unified Decoder Library Init .. DONE
Found 1 UTF Streams

2023/03/13 08:57:34.084609935 {iosrp_R0-0}{255}: [parser_cmd] [3793]: (note): id=
10.68.219.145@vty0:user= cmd: 'show logging profile wireless level info' SUCCESS 2023/03/13
 08:57:31.376 UTC
2023/03/13 09:07:03.562290152 {iosrp_R0-0}{255}: [parser_cmd] [3793]: (note): id=
10.68.219.145@vty0:user= cmd: 'show logging profile wireless internal ' SUCCESS 2023/03/13
 08:58:51.922 UTC
============================================================
======= Unified Trace Decoder Information/Statistics ======
============================================================
----------------- Decoder Input Information --------------
============================================================
```

```
 Num of Unique Streams .. 1
 Total UTF To Process ... 1
 Total UTM To Process ... 55410
 UTM Process Filter .....
qpp_cp,fman_fp,fman_rp,fman-rp,fman-fp,fed,mobilityd,mspd,odm_proxy,rogued,rrm,repm,wncd,wncmgrd,wncd_x,IOSRP,snd,odm,wstatsd,linux-iosd-image,wncloudm,locationd

 MRST Filter Rules ...... 24
============================================================
----------------- Decoder Output Information --------------
============================================================
 First UTM TimeStamp ............... 2023/03/13 08:13:19.321653302
 Last UTM TimeStamp ................ 2023/03/13 09:07:08.462269864
 UTM [Skipped / Rendered / Total] .. 55408 / 2 / 55410
 UTM [ENCODED] ..................... 2
 UTM [PLAIN TEXT] .................. 0
 UTM [DYN LIB] ..................... 0
 UTM [MODULE ID] ................... 0
 UTM [TDL TAN] ..................... 0
 UTM [APP CONTEXT] ................. 0
 UTM [MARKER] ...................... 0
 UTM [PCAP] ........................ 0
 UTM [LUID NOT FOUND] .............. 0
 UTM Level [EMERGENCY / ALERT / CRITICAL / ERROR] .. 0 / 0 / 0 / 0
 UTM Level [WARNING / NOTICE / INFO / DEBUG] ....... 0 / 2 / 0 / 0
 UTM Level [VERBOSE / NOISE / INVALID] ............. 0 / 0 / 0
============================================================
```

The following is an example of the **show logging process-helper** command.

```
Device# show logging process-helper ios
Logging display requested on 2023/03/13 10:30:29 (UTC) for Hostname: [FABRIEK], Model:
[C8300-1N1S-4T2X], Version: [X.X.X], SN: [FDO24190V85], MD_SN: [FDO2451M13G]

Displaying logs from the last 0 days, 0 hours, 10 minutes, 0 seconds
executing cmd on chassis local ...
Unified Decoder Library Init .. DONE
Found 1 UTF Streams

2023/03/13 10:30:16.884663022 {iosrp_R0-0}{255}: [parser_cmd] [3793]: (note): id=
10.68.219.145@vty0:user= cmd: 'enable' SUCCESS 2023/03/13 10:30:10.721 UTC
============================================================
======= Unified Trace Decoder Information/Statistics ======
============================================================
----------------- Decoder Input Information ---------------
============================================================
 Num of Unique Streams .. 1
 Total UTF To Process ... 1
 Total UTM To Process ... 88985
 UTM Process Filter ..... IOSRP
 MRST Filter Rules ...... 1
============================================================
----------------- Decoder Output Information --------------
============================================================
 First UTM TimeStamp ............... 2023/03/13 08:13:19.321653302
 Last UTM TimeStamp ................ 2023/03/13 10:30:27.267645695
 UTM [Skipped / Rendered / Total] .. 88984 / 1 / 88985
 UTM [ENCODED] ..................... 1
 UTM [PLAIN TEXT] .................. 0
 UTM [DYN LIB] ..................... 0
 UTM [MODULE ID] ................... 0
 UTM [TDL TAN] ..................... 0
 UTM [APP CONTEXT] ................. 0
 UTM [MARKER] ...................... 0
```

```
 UTM [PCAP] ........................ 0
 UTM [LUID NOT FOUND] .............. 0
 UTM Level [EMERGENCY / ALERT / CRITICAL / ERROR] .. 0 / 0 / 0 / 0
 UTM Level [WARNING / NOTICE / INFO / DEBUG] ....... 0 / 1 / 0 / 0
 UTM Level [VERBOSE / NOISE / INVALID] ............. 0 / 0 / 0
==========================================================
```

CHAPTER 149

Consent Token

Restrictions for Consent Token

- Consent Token is enabled by default and cannot be disabled.
- After the challenge has been sent from the device, the response needs to be entered within 30 minutes. If it is not entered, the challenge expires and a new challenge must be requested.
- A single response is valid only for one time for a corresponding challenge.
- The maximum authorization timeout for root-shell access is seven days.
- After a switchover event, all the existing Consent Token based authorizations would be treated as expired. You must then restart a fresh authentication sequence for service access.
- Only Cisco authorized personnel have access to Consent Token response generation on Cisco's challenge signing server.
- In System Shell access scenario, exiting the shell does not terminate authorization until the authorization timeout occurs or the shell authorization is explicitly terminated by the consent token terminate authorization command.

  We recommend that you force terminate System Shell authorization by explicitly issuing the Consent Token terminate command once the purpose of System Shell access is complete.

Information About Consent Token

Consent Token is a security feature that is used to authenticate the network administrator of an organization to access system shell with mutual consent from the network administrator and Cisco Technical Assistance Centre (Cisco TAC).

In some debugging scenarios, the Cisco TAC engineer may have to collect certain debug information or perform live debug on a production system. In such cases, the Cisco TAC engineer will ask you (the network

administrator) to access system shell on your device. Consent Token is a lock, unlock and re-lock mechanism that provides you with privileged, restricted, and secure access to the system shell.

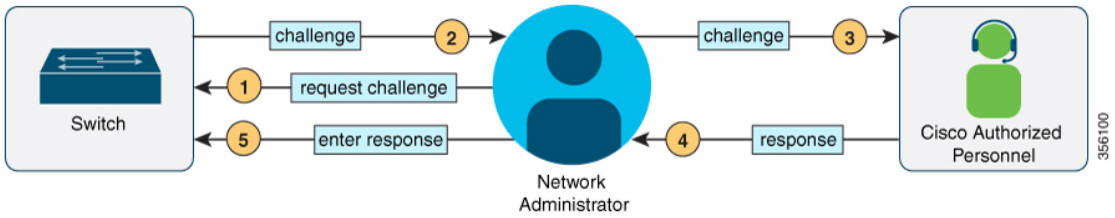
When you request access to system shell, you need to be authorized. You must first run the command to generate a challenge using the Consent Token feature on your device. The device generates a unique challenge as output. You must then copy this challenge string and send it to a Cisco Authorized Personnel through e-mail or Instant Message.

The Cisco Authorized Personnel processes the unique challenge string and generates a response that is unique. The Cisco Authorized Personnel copies this response string and sends it to you through e-mail or Instant Message.

You must then input this response string into your device. If the challenge-response pair match, you are authorized to access system shell. If not, an error is displayed and you are required to repeat the authentication process.

Once you gain access to system shell, collect the debug information required by the Cisco TAC engineer. After you are done accessing system shell, terminate the session and continue the debugging process.

Figure 143: Consent Token



Consent Token Authorization Process for System Shell Access

This section describes the process of Consent Token authorization to access system shell:

Procedure

Step 1 Generate a challenge requesting for access to system shell for the specified time period.

Example:

```
Device# request consent-token generate-challenge shell-access auth-timeout 900
zSSdrAAAAQEBAAQAAAABAgAEAAAAAAMACH86csUhmDl0BAAQ0Fvd7CxqRYUeoD7B4AwW7QUABAAAAG8GAAhDVEFf
REVNTwcAGENUQV9ERU1PX0NUQV9TSUdOSU5HX0tFWQgAC0M5ODAwLUNMLUs5CQALOVpQUEVESE5KRkI=
Device#
*Jan 18 02:47:06.733: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (challenge generation
attempt: Shell access 0).
```

Send a request for a challenge using the **request consent-token generate-challenge shell-access** *time-validity-slot* command. The duration in minutes for which you are requesting access to system shell is the time-slot-period.

In this example, the time period is 900 minutes after which the session expires.

The device generates a unique challenge as output. This challenge is a base-64 format string.

Step 2 Send the challenge string to a Cisco Authorized Personnel.

Send the challenge string generated by the device to a Cisco Authorized Personnel through e-mail or Instant Message.

The Cisco Authorized Personnel processes the unique challenge string and generates a response. The response is also a base-64 string that is unique. The Cisco Authorized Personnel copies this response string and sends it to you through e-mail or Instant Message.

Step 3 Input the response string onto your device.

Example:

```
Device# request consent-token accept-response shell-access
lR1y2AAUKFcAAAABAAABYlVDZ2d6MnkxL3JxTTJSSC9FZE5aW
nRSa1FteS9POWZqRUNlTk1LL3VoUWkIc0FsOHl5OW5vckQ4YWVOelpZZGYNCkNpWHY0b1B4Q000UGs1M2ZEMUpoazBCUkYyM3FML1A2ckVjM3paR05wd
Hcvck95UVduYUVufnA5bnhIZ09CNE0NCjBmVjh4b3I4TzE3aHNMaU1JeDQ3YWtkdE9Xb0JhImlzMVRweFBVZE93QUxvZDVEbmo4UEtiR01VVUM5b3lZ
WXQNCjFIRnJPbXczcmpsZTJHUnIxOWJUNkZLIWlpZ0ZmbENVRWo4K2xoaXgxS0ZtdDVPcDBYczVPSU43L0dSK1pGInoNCmYxIUtjaWlOWDhWTlNLQ0Z
WNURHU3pIenF1UFBxZVNDU0xLNkhXUTFROTlFMXJVakdlZ1NqIWxnNF1ySkJYL0wNCnpaTDVVRnVFdWpRWDdDUThIdkVPM1E9PQ==
% Consent token authorization success
*Jan 18 02:51:37.807: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (authentication success:
 Shell access 0).

Device# request platform software system shell
Activity within this shell can jeopardize the functioning of the system.
Are you sure you want to continue? [y/n] y
Device#
*Jan 18 02:56:59.714: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (authorization for Shell
 access 0 will expire in 10 min).
```

Input the response string sent to you by the Cisco Authorized Personnel using the **request consent-token accept-response shell-access** *response-string* command.

If the challenge-response pair match, you are authorized to access system shell. If the challenge-response pair do not match, an error is displayed and you are required to repeat steps 1 to 3.

After you are authorized, you can access system shell for the requested time-slot.

The device sends a message when there is ten minutes remaining of the authorization session.

Step 4 Terminate the session.

Example:

```
Device# request consent-token terminate-auth
% Consent token authorization termination success

Device#
*Jan 18 23:33:02.937: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (terminate authentication:
 Shell access 0).
Device#
```

When you finish accessing system shell, you can end the session using the **request consent-token terminate-auth** command. You can also force terminate the session prior to the authorization timeout using this command. The session also gets terminated automatically when the requested time slot expires.

CHAPTER 150

Troubleshooting the Software Configuration

This chapter describes how to identify and resolve software problems related to the Cisco IOS software on the switch. Depending on the nature of the problem, you can use the command-line interface (CLI), Device Manager, or Network Assistant to identify and solve problems.

Additional troubleshooting information, such as LED descriptions, is provided in the hardware installation guide.

Information About Troubleshooting the Software Configuration

Software Failure on a Switch

Switch software can be corrupted during an upgrade by downloading the incorrect file to the switch, and by deleting the image file. In all of these cases, there is no connectivity.

Lost or Forgotten Password on a Device

The default configuration for the device allows an end user with physical access to the device to recover from a lost password by interrupting the boot process during power-on and by entering a new password. These recovery procedures require that you have physical access to the device.

Note On these devices, a system administrator can disable some of the functionality of this feature by allowing an end user to reset a password only by agreeing to return to the default configuration. If you are an end user trying to reset a password when password recovery has been disabled, a status message reminds you to return to the default configuration during the recovery process.

Note You cannot recover encryption password key, when Cisco WLC configuration is copied from one Cisco WLC to another (in case of an RMA).

Follow the steps described in the section Recovering from a Lost or Forgotten Password, on page 2210 to recover from a lost or forgotten password.

Ping

The device supports IP ping, which you can use to test connectivity to remote hosts. Ping sends an echo request packet to an address and waits for a reply. Ping returns one of these responses:

- Normal response—The normal response (*hostname* is alive) occurs in 1 to 10 seconds, depending on network traffic.
- Destination does not respond—If the host does not respond, a *no-answer* message is returned.
- Unknown host—If the host does not exist, an *unknown host* message is returned.
- Destination unreachable—If the default gateway cannot reach the specified network, a *destination-unreachable* message is returned.
- Network or host unreachable—If there is no entry in the route table for the host or network, a *network or host unreachable* message is returned.

Refere the section Executing Ping, on page 2215 to understand how **ping** works.

Layer 2 Traceroute

The Layer 2 traceroute feature allows the switch to identify the physical path that a packet takes from a source device to a destination device. Layer 2 traceroute supports only unicast source and destination MAC addresses. Traceroute finds the path by using the MAC address tables of the devices in the path. When the Device detects a device in the path that does not support Layer 2 traceroute, the Device continues to send Layer 2 trace queries and lets them time out.

The Device can only identify the path from the source device to the destination device. It cannot identify the path that a packet takes from source host to the source device or from the destination device to the destination host.

Layer 2 Traceroute Guidelines

- Cisco Discovery Protocol (CDP) must be enabled on all the devices in the network. For Layer 2 traceroute to function properly, do not disable CDP.

  If any devices in the physical path are transparent to CDP, the switch cannot identify the path through these devices.

- A device is reachable from another device when you can test connectivity by using the **ping** privileged EXEC command. All devices in the physical path must be reachable from each other.
- The maximum number of hops identified in the path is ten.

- You can enter the **traceroute mac** or the **traceroute mac ip** privileged EXEC command on a device that is not in the physical path from the source device to the destination device. All devices in the path must be reachable from this switch.
- The **traceroute mac** command output shows the Layer 2 path only when the specified source and destination MAC addresses belong to the same VLAN. If you specify source and destination MAC addresses that belong to different VLANs, the Layer 2 path is not identified, and an error message appears.
- If you specify a multicast source or destination MAC address, the path is not identified, and an error message appears.
- If the source or destination MAC address belongs to multiple VLANs, you must specify the VLAN to which both the source and destination MAC addresses belong. If the VLAN is not specified, the path is not identified, and an error message appears.
- The **traceroute mac ip** command output shows the Layer 2 path when the specified source and destination IP addresses belong to the same subnet. When you specify the IP addresses, the device uses the Address Resolution Protocol (ARP) to associate the IP addresses with the corresponding MAC addresses and the VLAN IDs.
  - If an ARP entry exists for the specified IP address, the device uses the associated MAC address and identifies the physical path.
  - If an ARP entry does not exist, the device sends an ARP query and tries to resolve the IP address. If the IP address is not resolved, the path is not identified, and an error message appears.
- When multiple devices are attached to one port through hubs (for example, multiple CDP neighbors are detected on a port), the Layer 2 traceroute feature is not supported. When more than one CDP neighbor is detected on a port, the Layer 2 path is not identified, and an error message appears.
- This feature is not supported in Token Ring VLANs.
- Layer 2 traceroute opens a listening socket on the User Datagram Protocol (UDP) port 2228 that can be accessed remotely with any IPv4 address, and does not require any authentication. This UDP socket allows to read VLAN information, links, presence of particular MAC addresses, and CDP neighbor information, from the device. This information can be used to eventually build a complete picture of the Layer 2 network topology.
- Layer 2 traceroute is enabled by default and can be disabled by running the **no l2 traceroute** command in global configuration mode. To re-enable Layer 2 traceroute, use the **l2 traceroute** command in global configuration mode.

IP Traceroute

You can use IP traceroute to identify the path that packets take through the network on a hop-by-hop basis. The command output displays all network layer (Layer 3) devices, such as routers, that the traffic passes through on the way to the destination.

Your Device can participate as the source or destination of the **traceroute** privileged EXEC command and might or might not appear as a hop in the **traceroute** command output. If the Device is the destination of the traceroute, it is displayed as the final destination in the traceroute output. Intermediate devices do not show up in the traceroute output if they are only bridging the packet from one port to another within the same VLAN. However, if the intermediate Device is a multilayer Device that is routing a particular packet, this device shows up as a hop in the traceroute output.

The **traceroute** privileged EXEC command uses the Time To Live (TTL) field in the IP header to cause routers and servers to generate specific return messages. Traceroute starts by sending a User Datagram Protocol (UDP) datagram to the destination host with the TTL field set to 1. If a router finds a TTL value of 1 or 0, it drops the datagram and sends an Internet Control Message Protocol (ICMP) time-to-live-exceeded message to the sender. Traceroute finds the address of the first hop by examining the source address field of the ICMP time-to-live-exceeded message.

To identify the next hop, traceroute sends a UDP packet with a TTL value of 2. The first router decrements the TTL field by 1 and sends the datagram to the next router. The second router sees a TTL value of 1, discards the datagram, and returns the time-to-live-exceeded message to the source. This process continues until the TTL is incremented to a value large enough for the datagram to reach the destination host (or until the maximum TTL is reached).

To learn when a datagram reaches its destination, traceroute sets the UDP destination port number in the datagram to a very large value that the destination host is unlikely to be using. When a host receives a datagram destined to itself containing a destination port number that is unused locally, it sends an ICMP *port-unreachable* error to the source. Because all errors except port-unreachable errors come from intermediate hops, the receipt of a port-unreachable error means that this message was sent by the destination port.

Go to Example: Performing a Traceroute to an IP Host, on page 2221 to see an example of IP traceroute process.

Debug Commands

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. It is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

All **debug** commands are entered in privileged EXEC mode, and most **debug** commands take no arguments.

System Report

System reports or crashinfo files save information that helps Cisco technical support representatives to debug problems that caused the Cisco IOS image to fail (crash). It is necessary to quickly and reliably collect critical crash information with high fidelity and integrity. Further, it is necessary to collect this information and bundle it in a way that it can be associated or identified with a specific crash occurrence.

System reports are generated in case of a switchover: System reports are generated only on high availability (HA) member switches. Reports are not generated for non-HA members.

The system does not generate reports in case of a reload.

During a process crash, the following is collected locally from the switch:

1. Full process core
2. Tracelogs
3. IOS syslogs (not guaranteed in case of non-active crashes)
4. System process information

5. Bootup logs
6. Reload logs
7. Certain types of /proc information

This information is stored in separate files which are then archived and compressed into one bundle. This makes it convenient to get a crash snapshot in one place, and can be then moved off the box for analysis. This report is generated before the switch goes down to rommon/bootloader.

Except for the full core and tracelogs, everything else is a text file.

Use the **request platform software process core fed switch active** command to generate the core dump.

```
Device# request platform software process core fed switch active
SUCCESS: Core file generated.

Device# dir bootflash:/core
Directory of bootflash:/core/
16430   -rw-         10941657   Apr 6 2022 00:15:20 +00:00
Switch_1_RP_0_fed_18469_20220406-001511-UTC.core.gz
16812   -rw-               1   Apr 6 2022 00:01:48 +00:00  .callhome
16810   drwx            4096  Jan 18 2022 21:10:35 +00:00  modules
```

Crashinfo Files

By default the system report file will be generated and saved into the /crashinfo directory. Ifit cannot be saved to the crashinfo partition for lack of space, then it will be saved to the /flash directory.

To display the files, enter the **dir crashinfo:** command. The following is sample output of a crashinfo directory:

System reports are located in the crashinfo directory in the following format:

```
system-report_[switch number]_[date]-[timestamp]-UTC.gz
```

After a switch crashes, check for a system report file. The name of the most recently generated system report file is stored in the last_systemreport file under the crashinfo directory. The system report and crashinfo files assist TAC while troubleshooting the issue.

The system report generated can be further copied using TFTP, HTTP and few other options.

```
Device# copy crashinfo: ?
crashinfo:     Copy to crashinfo: file system
flash:         Copy to flash: file system
ftp:           Copy to ftp: file system
http:          Copy to http: file system
https:         Copy to https: file system
null:          Copy to null: file system
nvram:         Copy to nvram: file system
rcp:           Copy to rcp: file system
running-config Update (merge with) current system configuration
scp:           Copy to scp: file system
startup-config Copy to startup configuration
syslog:        Copy to syslog: file system
system:        Copy to system: file system
tftp:          Copy to tftp: file system
tmpsys:        Copy to tmpsys: file system
```

The general syntax for copying onto TFTP server is as follows:

```
Device# copy crashinfo: tftp:
Source filename [system-report_1_20150909-092728-UTC.gz]?
Address or name of remote host []? 1.1.1.1
Destination filename [system-report_1_20150909-092728-UTC.gz]?
```

The tracelogs can be collected by issuing a trace archive command. This command provides time period options. The command syntax is as follows:

```
Device# request platform software trace archive ?
last     Archive trace files of last x days
target   Location and name for the archive file
```

The tracelogs stored in crashinfo: or flash: directory from within the last 3650 days can be collected.

```
Device# request platform software trace archive last ?
<1-3650> Number of days (1-3650)
Device# request platform software trace archive last 3650 days target ?
crashinfo: Archive file name and location
flash:     Archive file name and location
```

Note It is important to clear the system reports or trace archives from flash or crashinfo directory once they are copied out, in order to have space available for tracelogs and other purposes.

In a complex network it is difficult to track the origin of a system-report file. This task is made easier if the system-report files are uniquely identifiable. The hostname will be prepended to the system-report file name making the reports uniquely identifiable.

The following example displays system-report files with the hostname prepended:

```
HOSTNAME# dir flash:/core | grep HOSTNAME
40486  -rw-        108268293  Oct 21 2019 16:07:50 -04:00
HOSTNAME-system-report_20191021-200748-UTC.tar.gz
40487  -rw-            17523  Oct 21 2019 16:07:56 -04:00
HOSTNAME-system-report_20191021-200748-UTC-info.txt
40484  -rw-         48360998  Oct 21 2019 16:55:24 -04:00
HOSTNAME-system-report_20191021-205523-UTC.tar.gz
40488  -rw-            14073  Oct 21 2019 16:55:26 -04:00
HOSTNAME-system-report_20191021-205523-UTC-info.txt
```

Onboard Failure Logging on the Switch

You can use the onboard failure logging (OBFL) feature to collect information about the device. The information includes uptime, temperature, and voltage information and helps Cisco technical support representatives to troubleshoot device problems. We recommend that you keep OBFL enabled and do not erase the data stored in the flash memory.

By default, OBFL is enabled. It collects information about the device and small form-factor pluggable (SFP) modules. The device stores this information in the flash memory:

- CLI commands—Record of the OBFL CLI commands that are entered on a standalone device.
- Message—Record of the hardware-related system messages generated by a standalone device .
- Temperature—Temperature of a standalone device.
- Uptime data—Time when a standalone device starts, the reason the device restarts, and the length of time the device has been running since it last restarted.
- Voltage—System voltages of a standalone device.

You should manually set the system clock or configure it by using Network Time Protocol (NTP).

When the device is running, you can retrieve the OBFL data by using the **show logging onboard** privileged EXEC commands. If the device fails, contact your Cisco technical support representative to find out how to retrieve the data.

When an OBFL-enabled device is restarted, there is a 10-minute delay before logging of new data begins.

Possible Symptoms of High CPU Utilization

Excessive CPU utilization might result in these symptoms, but the symptoms might also result from other causes, some of which are the following:

- Spanning tree topology changes
- EtherChannel links brought down due to loss of communication
- Failure to respond to management requests (ICMP ping, SNMP timeouts, slow Telnet or SSH sessions)
- UDLD flapping
- IP SLAs failures because of SLAs responses beyond an acceptable threshold
- DHCP or IEEE 802.1x failures if the switch does not forward or respond to requests

How to Troubleshoot the Software Configuration

Booting from the Recovery Partition

The switch support booting from the recovery partition. This is beneficial to end users if they face an issue while trying to boot the switch from Flash or an external device, such as USB or SDflash. The recovery image is the same as the recommended Cisco IOS image for the switch, and is stored in a partition named **drec0**.

Note You can not access recovery partition when the switch is in Cisco IOS prompt. Note that the factory-reset process does not erase this image.

To check the partition image name, enter **dir drec0:**

```
switch: dir drec0:

Attributes        Size         Name

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

-rw-r--r--   490586943   ie35xx-17.17.1.SPA.bin

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

switch:
```

To boot from the recovery partition, enter **boot drec0:<image name>**:

```
switch: boot drec0:ie35xx_lite_iosxe.17.17.01.SPA.bin

boot: attempting to boot from [drec0:ie35xx-17.17.1.SPA.bin]
```

```
boot: reading file ie35xx-17.17.1.SPA.bin
##############################
```

Recovering from a Lost or Forgotten Password

The default configuration for the switch allows an end user with physical access to the switch to recover from a lost password by interrupting the boot process during power-on and by entering a new password. These recovery procedures require that you have physical access to the switch.

Note On these switches, a system administrator can disable some of the functionality of this feature by allowing an end user to reset a password only by agreeing to return to the default configuration. If you are an end user trying to reset a password when password recovery has been disabled, a status message shows this during the recovery process.

Procedure

Step 1 Connect a terminal or PC to the switch.

- Connect a terminal or a PC with terminal-emulation software to the switch console port.
- Connect a PC to the Ethernet management port.

Step 2 Set the line speed on the emulation software to 9600 baud.

Step 3 Power off the standalone switch.

Step 4 Reconnect the power cord to the switch or the active switch. As soon as the System LED blinks, press and release the Mode button 2-3 times. The switch enters the ROMMON mode.

The following console messages are displayed during the reload:

```
Initializing Hardware...

System Bootstrap, Version xx.x.1r [FC1], RELEASE SOFTWARE (P)
Compiled Tue 10/29/2024 18:05:06 by rel

Current ROMMON image : Primary
IE35xx-24P platform with 4194304 Kbytes of main memory

Preparing to autoboot. [Press Ctrl-C to interrupt]  4    (interrupted) <-------- break
sequence to be pressed
```

Proceed to the *Procedure with Password Recovery Enabled* section, and follow the steps.

Step 5 After recovering the password, reload the switch or the active switch.

On a switch:

```
Switch> reload
Proceed with reload? [confirm] y
```

Procedure with Password Recovery Enabled

Procedure

Step 1 Enable manual boot mode.

```
Device: MANUAL_BOOT=yes
```

Step 2 Ignore the startup configuration with the following command:

```
Device: SWITCH_IGNORE_STARTUP_CFG=1
```

Step 3 Boot the switch with the *packages.conf* file from flash.

```
Device: boot flash:packages.conf
```

Step 4 Terminate the initial configuration dialog by answering **No**.

```
Would you like to enter the initial configuration dialog? [yes/no]: No
```

Step 5 At the switch prompt, enter privileged EXEC mode.

```
Device> enable
Device#
```

Step 6 Copy the startup configuration to running configuration.

```
Device# copy startup-config running-config Destination filename [running-config]?
```

Press Return in response to the confirmation prompts. The configuration file is now reloaded, and you can change the password.

Step 7 Enter global configuration mode and change the **enable** password.

```
Device# configure terminal
Device(config)# enable secret password
```

Step 8 Set the SWITCH_IGNORE_STARTUP_CFG parameter to 0.

```
Device(config)# no system ignore startupconfig switch all
Device(config)# end
```

Step 9 Write the running configuration to the startup configuration file and save the configuration.

```
Device# copy running-config startup-config
```

```
Device# write memory
```

Step 10 Confirm that manual boot mode is enabled.

```
Device# show boot

 BOOT variable = flash:packages.conf;
 Manual Boot = yes
 Enable Break = yes
```

Step 11 Reload the device.

```
Device# reload
```

Step 12 Boot the device with the *packages.conf* file from flash.

```
Device: boot flash:packages.conf
```

Step 13 After the device boots up, disable manual boot on the device.

```
Device(config)# no boot manual
```

Procedure with Password Recovery Disabled

If the password-recovery mechanism is disabled, this message appears:

```
The password-recovery mechanism has been triggered, but
is currently disabled.  Access to the boot loader prompt
through the password-recovery mechanism is disallowed at
this point.  However, if you agree to let the system be
reset back to the default system configuration, access
to the boot loader prompt can still be allowed.

Would you like to reset the system back to the default configuration (y/n)?
```

Caution Returning the device to the default configuration results in the loss of all existing configurations. We recommend that you contact your system administrator to verify if there are backup device and VLAN configuration files.

- If you enter **n** (no), the normal boot process continues as if the **Mode** button had not been pressed; you cannot access the boot loader prompt, and you cannot enter a new password. You see the message:

  ```
  Press Enter to continue........
  ```

- If you enter **y** (yes), the configuration file in flash memory and the VLAN database file are deleted. When the default configuration loads, you can reset the password.

Procedure

Step 1 Choose to continue with password recovery and delete the existing configuration:

```
Would you like to reset the system back to the default configuration (y/n)? Y
```

Step 2 Display the contents of flash memory:

```
Device: dir flash:
```

The device file system appears.

Step 3 Boot up the system:

```
Device: boot
```

You are prompted to start the setup program. To continue with password recovery, enter **N** at the prompt:

```
Continue with the configuration dialog? [yes/no]: N
```

Step 4 At the device prompt, enter privileged EXEC mode:

```
Device> enable
```

Step 5 Enter global configuration mode:

```
Device# configure terminal
```

Step 6 Change the password:

```
Device(config)# enable secret password
```

The secret password can be from 1 to 25 alphanumeric characters, can start with a number, is case sensitive, and allows spaces but ignores leading spaces.

Step 7 Return to privileged EXEC mode:

```
Device(config)# exit
Device#
```

Step 8 Write the running configuration to the startup configuration file:

```
Device# copy running-config startup-config
```

The new password is now in the startup configuration.

Step 9 You must now reconfigure the device. If the system administrator has the backup device and VLAN configuration files available, you should use those.

Preventing Autonegotiation Mismatches

The IEEE 802.3ab autonegotiation protocol manages the device settings for speed (10 Mb/s, 100 Mb/s, and 1000 Mb/s, excluding SFP module ports) and duplex (half or full). There are situations when this protocol can incorrectly align these settings, reducing performance. A mismatch occurs under these circumstances:

- A manually set speed or duplex parameter is different from the manually set speed or duplex parameter on the connected port.
- A port is set to autonegotiate, and the connected port is set to full duplex with no autonegotiation.

To maximize the device performance and ensure a link, follow one of these guidelines when changing the settings for duplex and speed:

- Let both ports autonegotiate both speed and duplex.
- Manually set the speed and duplex parameters for the ports on both ends of the connection.

Note If a remote device does not autonegotiate, configure the duplex settings on the two ports to match. The speed parameter can adjust itself even if the connected port does not autonegotiate.

Troubleshooting SFP Module Security and Identification

Cisco small form-factor pluggable (SFP) modules have a serial EEPROM that contains the module serial number, the vendor name and ID, a unique security code, and cyclic redundancy check (CRC). When an SFP module is inserted in the device, the device software reads the EEPROM to verify the serial number, vendor name and vendor ID, and recompute the security code and CRC. If the serial number, the vendor name or vendor ID, the security code, or CRC is invalid, the software generates a security error message and places the interface in an error-disabled state.

Note The security error message references the GBIC_SECURITY facility. The device supports SFP modules and does not support GBIC modules. Although the error message text refers to GBIC interfaces and modules, the security messages actually refer to the SFP modules and module interfaces.

If you are using a non-Cisco SFP module, remove the SFP module from the device, and replace it with a Cisco module. After inserting a Cisco SFP module, use the **errdisable recovery cause gbic-invalid** global configuration command to verify the port status, and enter a time interval for recovering from the error-disabled state. After the elapsed interval, the device brings the interface out of the error-disabled state and retries the operation. For more information about the **errdisable recovery** command, see the command reference for this release.

If the module is identified as a Cisco SFP module, but the system is unable to read vendor-data information to verify its accuracy, an SFP module error message is generated. In this case, you should remove and reinsert the SFP module. If it continues to fail, the SFP module might be defective.

Executing Ping

If you attempt to ping a host in a different IP subnetwork, you must define a static route to the network or have IP routing configured to route between those subnets.

IP routing is disabled by default on all devices.

Note Though other protocol keywords are available with the **ping** command, they are not supported in this release.

Use this command to ping another device on the network from the device:

Command	Purpose
ping ip *host* | *address* `Device# ping 172.20.52.3`	Pings a remote host through IP or by supplying the hostname or network address.

Monitoring the Physical Path

You can monitor the physical path that a packet takes from a source device to a destination device by using one of these privileged EXEC commands:

Table 153: Monitoring the Physical Path

Command	Purpose
tracetroute mac [**interface** *interface-id*] {*source-mac-address*} [**interface** *interface-id*] {*destination-mac-address*} [**vlan** *vlan-id*] [**detail**]	Displays the Layer 2 path taken by the packets from the specified source MAC address to the specified destination MAC address.
tracetroute mac ip {*source-ip-address* | *source-hostname*} {*destination-ip-address* | *destination-hostname*} [**detail**]	Displays the Layer 2 path taken by the packets from the specified source IP address or hostname to the specified destination IP address or hostname.

Executing IP Traceroute

Note Though other protocol keywords are available with the **traceroute** privileged EXEC command, they are not supported in this release.

Command	Purpose
traceroute ip *host* `Device# traceroute ip 192.51.100.1`	Traces the path that packets take through the network.

Redirecting Debug and Error Message Output

By default, the network server sends the output from **debug** commands and system error messages to the console. If you use this default, you can use a virtual terminal connection to monitor debug output instead of connecting to the console port .

Possible destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server. The syslog format is compatible with 4.3 Berkeley Standard Distribution (BSD) UNIX and its derivatives.

Note Be aware that the debugging destination you use affects system overhead. When you log messages to the console, very high overhead occurs. When you log messages to a virtual terminal, less overhead occurs. Logging messages to a syslog server produces even less, and logging to an internal buffer produces the least overhead of any method.

For more information about system message logging, see *Configuring System Message Logging*.

Using the show platform Command

The output from the **show platform** privileged EXEC command provides some useful information about the forwarding results if a packet entering an interface is sent through the system. Depending upon the parameters entered about the packet, the output provides lookup table results and port maps used to calculate forwarding destinations, bitmaps, and egress information.

Most of the information in the output from the command is useful mainly for technical support personnel, who have access to detailed information about the device application-specific integrated circuits (ASICs). However, packet forwarding information can also be helpful in troubleshooting.

Using the show debug command

The **show debug** command is entered in privileged EXEC mode. This command displays all debug options available on the switch.

To view all conditional debug options run the command **show debug condition** The commands can be listed by selecting either a condition identifier *<1-1000>* or *all* conditions.

To disable debugging, use the **no debug all** command.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Verifying Troubleshooting of the Software Configuration

Displaying OBFL Information

Example: Verifying the Problem and Cause for High CPU Utilization

To determine if high CPU utilization is a problem, enter the **show processes cpu sorted** privileged EXEC command. Note the underlined information in the first line of the output example.

```
Device# show processes cpu sorted
CPU utilization for five seconds: 8%/0%; one minute: 7%; five minutes: 8%
PID Runtime(ms) Invoked uSecs 5Sec 1Min 5Min TTY Process
309 42289103 752750 56180 1.75% 1.20% 1.22% 0 RIP Timers
140 8820183 4942081 1784 0.63% 0.37% 0.30% 0 HRPC qos request
100 3427318 16150534 212 0.47% 0.14% 0.11% 0 HRPC pm-counters
192 3093252 14081112 219 0.31% 0.14% 0.11% 0 Spanning Tree
143 8 37 216 0.15% 0.01% 0.00% 0 Exec
...
<output truncated>
```

This example shows normal CPU utilization. The output shows that utilization for the last 5 seconds is 8%/0%, which has this meaning:

- The total CPU utilization is 8 percent, including both time running Cisco IOS processes and time spent handling interrupts.
- The time spent handling interrupts is zero percent.

Table 154: Troubleshooting CPU Utilization Problems

Type of Problem	Cause	Corrective Action
Interrupt percentage value is almost as high as total CPU utilization value.	The CPU is receiving too many packets from the network.	Determine the source of the network packet. Stop the flow, or change the switch configuration. See the section on "Analyzing Network Traffic."
Total CPU utilization is greater than 50% with minimal time spent on interrupts.	One or more Cisco IOS process is consuming too much CPU time. This is usually triggered by an event that activated the process.	Identify the unusual event, and troubleshoot the root cause. See the section on "Debugging Active Processes."

Scenarios for Troubleshooting the Software Configuration

Scenarios to Troubleshoot Power over Ethernet (PoE)

Table 155: Power over Ethernet Troubleshooting Scenarios

Symptom or Problem	Possible Cause and Solution
Only one port does not have PoE. Trouble is on only one switch port. PoE and non-PoE devices do not work on this port, but do on other ports.	Verify that the powered device works on another PoE port. Use the **show run**, or **show interface status** user EXEC commands to verify that the port is not shut down or error-disabled. **Note** Most switches turn off port power when the port is shut down, even though the IEEE specifications make this optional. Verify that **power inline never** is not configured on that interface or port. Verify that the Ethernet cable from the powered device to the switch port is good: Connect a known good non-PoE Ethernet device to the Ethernet cable, and make sure that the powered device establishes a link and exchanges traffic with another host. **Note** Cisco powered device works only with straight cable and not with crossover one. Verify that the total cable length from the switch front panel to the powered device is not more than 100 meters. Disconnect the Ethernet cable from the switch port. Use a short Ethernet cable to connect a known good Ethernet device directly to this port on the switch front panel (not on a patch panel). Verify that it can establish an Ethernet link and exchange traffic with another host, or ping the port VLAN SVI. Next, connect a powered device to this port, and verify that it powers on. If a powered device does not power on when connected with a patch cord to the switch port, compare the total number of connected powered devices to the switch power budget (available PoE). Use the **show power inline** command to verify the amount of available power.

Symptom or Problem	Possible Cause and Solution
No PoE on all ports or a group of ports. Trouble is on all switch ports. Nonpowered Ethernet devices cannot establish an Ethernet link on any port, and PoE devices do not power on.	If there is a continuous, intermittent, or reoccurring alarm related to power, replace the power supply if possible it is a field-replaceable unit. Otherwise, replace the switch. If the problem is on a consecutive group of ports but not all ports, the power supply is probably not defective, and the problem could be related to PoE regulators in the switch. Use the **show log** privileged EXEC command to review alarms or system messages that previously reported PoE conditions or status changes. If there are no alarms, use the **show interface status** command to verify that the ports are not shut down or error-disabled. If ports are error-disabled, use the **shut** and **no shut** interface configuration commands to reenable the ports. Use the **show env power** and **show power inline** privileged EXEC commands to review the PoE status and power budget (available PoE). Review the running configuration to verify that **power inline never** is not configured on the ports. Connect a nonpowered Ethernet device directly to a switch port. Use only a short patch cord. Do not use the existing distribution cables. Enter the **shut** and **no shut** interface configuration commands, and verify that an Ethernet link is established. If this connection is good, use a short patch cord to connect a powered device to this port and verify that it powers on. If the device powers on, verify that all intermediate patch panels are correctly connected. Disconnect all but one of the Ethernet cables from switch ports. Using a short patch cord, connect a powered device to only one PoE port. Verify the powered device does not require more power than can be delivered by the switch port. Use the **show power inline** privileged EXEC command to verify that the powered device can receive power when the port is not shut down. Alternatively, watch the powered device to verify that it powers on. If a powered device can power on when only one powered device is connected to the switch, enter the **shut** and **no shut** interface configuration commands on the remaining ports, and then reconnect the Ethernet cables one at a time to the switch PoE ports. Use the **show interface status** and **show power inline** privileged EXEC commands to monitor inline power statistics and port status. If there is still no PoE at any port, a fuse might be open in the PoE section of the power supply. This normally produces an alarm. Check the log again for alarms reported earlier by system messages.

Symptom or Problem	Possible Cause and Solution
IEEE 802.3af-compliant or IEEE 802.3at-compliant powered devices do not work on Cisco PoE switch. A non-Cisco powered device is connected to a Cisco PoE switch, but never powers on or powers on and then quickly powers off. Non-PoE devices work normally.	Use the **show power inline** command to verify that the switch power budget (available PoE) is not depleted before or after the powered device is connected. Verify that sufficient power is available for the powered device type before you connect it. Use the **show interface status** command to verify that the switch detects the connected powered device. Use the **show log** command to review system messages that reported an overcurrent condition on the port. Identify the symptom precisely: Does the powered device initially power on, but then disconnect? If so, the problem might be an initial surge-in (or *inrush*) current that exceeds a current-limit threshold for the port.

Configuration Examples for Troubleshooting Software

Example: Pinging an IP Host

This example shows how to ping an IP host:

```
Device# ping 172.20.52.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echoes to 172.20.52.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Device#
```

Table 156: Ping Output Display Characters

Character	Description
!	Each exclamation point means receipt of a reply.
.	Each period means the network server timed out while waiting for a reply.
U	A destination unreachable error PDU was received.
C	A congestion experienced packet was received.
I	User interrupted test.
?	Unknown packet type.
&	Packet lifetime exceeded.

To end a ping session, enter the escape sequence (**Ctrl-^ X** by default). Simultaneously press and release the **Ctrl**, **Shift**, and **6** keys and then press the **X** key.

Example: Performing a Traceroute to an IP Host

This example shows how to perform a **traceroute** to an IP host:

```
Device# traceroute ip 192.0.2.10

Type escape sequence to abort.
Tracing the route to 192.0.2.10

  1 192.0.2.1 0 msec 0 msec 4 msec
  2 192.0.2.203 12 msec 8 msec 0 msec
  3 192.0.2.100 4 msec 0 msec 0 msec
  4 192.0.2.10 0 msec 4 msec 0 msec
```

The display shows the hop count, the IP address of the router, and the round-trip time in milliseconds for each of the three probes that are sent.

Table 157: Traceroute Output Display Characters

Character	Description
*	The probe timed out.
?	Unknown packet type.
A	Administratively unreachable. Usually, this output means that an access list is blocking traffic.
H	Host unreachable.
N	Network unreachable.
P	Protocol unreachable.
Q	Source quench.
U	Port unreachable.

To end a trace in progress, enter the escape sequence (**Ctrl-^ X** by default). Simultaneously press and release the **Ctrl**, **Shift**, and **6** keys and then press the **X** key.

CHAPTER 151

Line Auto Consolidation

Line Auto Consolidation

Cisco IOS XE software runs a nonvolatile generation (NVGEN) process to retrieve the configuration state of the device. During the NVGEN process, the system auto consolidates the LINE commands based on common parameters.

When the device connects to Cisco Catalyst Center Center, the center sends a line configuration through the Yet Another Next Generation (YANG) interface the resulting configuration is auto consolidated. This can cause a mismatch between the device and the Cisco Catalyst Center. The mismatch in configurations can lead to reverse sync from the device to the Cisco Catalyst Center. The device will be locked from any other configuration changes during this reverse sync. This can affect the performance of the device.

You can use the **no line auto-consolidation** command, in the global configuration mode, to disable the auto consolidation of LINE commands. Auto consolidation is enabled by default. To disable it use the no form of the command.

You can use the **show running-configuration all** command to display the configuration on the device. In the following example line auto-consolidation is enabled.

```
Device#sh running-config all | i auto-consolidation
line auto-consolidation
```

After auto consolidation is disabled the **show run** command output will be lengthy. This will impact the sizes of the running configuration and start-up configuration files. If you disable auto consolidation you will observe the following behaviors:

- Contiguous groups of lines that belong to the same configuration in a sub-mode will not be combined into a single range.

```
Device#show run | sec line
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 9
transport input all
Device#configure terminal
Device(config)#no line auto-consolidation
Device(config)#line vty 10 15
Device(config-line)#transport input all
```

```
Device(config-line)#end
Device#show run | sec line
no line auto-consolidation
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 9
transport input all
line vty 10 15
transport input all
```

- If you disable auto consolidation after configuring some lines with auto consolidation enabled, only the lines which were configured after auto consolidation was disabled will not be consolidated.

```
Device#show run | sec line
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 9
transport input all
Device#configure terminal
Device(config)#line vty 10 15
Device(config-line)#transport input all
Device(config-line)#end
Device#show run | sec line
line con 0
stopbits 1
consolidated line vty 0 4
transport input ssh
line vty 5 15
transport input all
Device#configure terminal
Device(config)#no line auto-consolidation
Device(config)#line vty 16 20
Device(config-line)#transport input all
Device(config-line)#end
Device#show run | sec line
no line auto-consolidation
line con 0
stopbits 1
consolidated line vty 0 4
transport input ssh
line vty 5 15
transport input all
line vty 16 20
transport input all
```

- If you enable auto consolidation after it has been disabled, lines that were not consolidated will be auto consolidated.

```
Device#sh running-config | sec line
no line auto-consolidation
line con 0
exec-timeout 0 0
logging synchronous
stopbits 1
line vty 0 4
transport input ssh
line vty 5 15
transport input ssh
line vty 16 19
transport input ssh
```

```
Device#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)#line vty 20 25
Device(config-line)#transport input ssh
Device(config-line)#end
Device#sh running-config | sec line
no line auto-consolidation
line con 0
exec-timeout 0 0
logging synchronous
stopbits 1
line vty 0 4
transport input ssh
line vty 5 15
transport input ssh
line vty 16 19
transport input ssh
line vty 20 25
transport input ssh
Device#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)#line auto-consolidation
Device(config)#end
Device#show running-config | sec line
line con 0
exec-timeout 0 0
logging synchronous
stopbits 1
line vty 0 4
transport input ssh
line vty 5 25
transport input ssh
```

- You can configure lines with contiguous ranges. The configuration will be permitted.

```
Device#show run | sec line
no line auto-consolidation
line con 0
stopbits 1
line vty 0 4
transport input ssh
Device#configure terminal
Device(config)#line vty 5  20
Device(config)#transport input all
Device(config-line)#end
Device#show run | sec line
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 20
transport input all
```

- You can't configure lines with non-contiguous ranges. The configuration is rejected.

```
Device#show run | sec line
no line auto-consolidation
line con 0
logging synchronous
line aux 0
line vty 0 4
transport input none
Device# configure terminal
```

```
Device(config)# line vty 10 20
% Bad line number - VTY line number is not contiguous.
```

- You can delete lines which are contiguous and at the end of the list. In the controller mode, you can delete one line at a time. You cannot delete lines in bulk. In autonomous mode, you can delete lines in bulk.

```
Device# show run | sec line
no line auto-consolidation
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 20
transport input all
Device# configure terminal
Device(config)# no line vty 5 20
Device(config)# end
Device#show run | sec line
line con 0
stopbits 1
line vty 0 4
transport input ssh
```

- You can't delete lines which are not contiguous and at the end of the list. You can't delete a line that will result in a non-contiguous range when it is deleted. This will generate an error stating the line cannot be deleted.

```
Device# show run | sec line
no line auto-consolidation
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 9
transport input none
line vty 10 20
transport input all
Device# configure terminal
Device(config)# no line vty 5 9
% Cannot delete the 9 line number as it is not the last VTY line number
```

- You can't delete lines that are in use or are default lines.

```
Device#show run | sec line
no line auto-consolidation
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 20
transport input ssh
Device#configure terminal
Router(config)#no line vty 15
% Can't delete last 16 VTY lines, lines in use, statbit: 0x10C40, tiptop: 590
% process name: SSH Process
```

- You can modify subranges in autonomous mode. This will cause the lines to split which will cause a reverse sync of the configuration. You can’t modify subranges in the controller mode. This is a behavioural change between the controller and autonomous modes. In the controller mode, any modification of subranges is rejected to avoid discrepancy with the configuration pushed from a controller.

  The following examples shows how you can modify subranges in autonomous mode.

```
Device#show run | sec line
no line auto-consolidation
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 9
transport input none
Device#configure terminal
Device(config)#line vty 7 8
Device(config-line)#transport input telnet
Device(config-line)#end
Device#show run | sec line
line con 0
 stopbits 1
line vty 0 4
 transport input ssh
line vty 5 6
 transport input none
line vty 7 8
 transport input telnet
line vty 9
 transport input none
```

- The following example shows that modification of subranges is not supported in controller mode

```
Device#show run | sec line
no line auto-consolidation
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 9
transport input none
Device#configure terminal
Device(config)# line vty 5 8
Device(config-line)# end
Uncommitted changes found, commit them? [yes/no/CANCEL] yes
Aborted: inconsistent value: Device refused one or more commands:
line vty 5 8
      ^
% Invalid input detected at '^' marker.
Component Response: "
% Modifications of overlapping/sub range is not allowed in controller mode"
Error executing command: CLI command error -
Device(config)# end
```

- You can modify overlapping ranges in autonomous mode. This will cause the lines to split which will cause a reverse sync of the configuration. You cannot modify overlapping ranges in the controller mode. In the controller mode, any modification of overlapping ranges is rejected to avoid discrepancy with the configuration pushed from a controller.

  The following example shows how you can modify overlapping ranges in autonomous mode.

```
Device#show run | sec line
no line auto-consolidation
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 10
transport input none
line vty 11 20
transport input all
```

```
Device#configure terminal
Device(config)#line vty 8 12
Device(config-line)#transport input ssh
Device(config-line)#end
Device#show run | sec line
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 7
transport input none
line vty 8 10
transport input ssh
line vty 11 12
transport input ssh
line vty 13 20
transport input all
```

- The following example shows that modification of overlapping ranges is not supported in controller mode.

```
Device#show run | sec line
no line auto-consolidation
line con 0
stopbits 1
line vty 0 4
transport input ssh
line vty 5 10
transport input none
line vty 11 20
transport input all
Device(config)# line vty 5 11
Device(config-line)# end
Uncommitted changes found, commit them? [yes/no/CANCEL] yes
Aborted: inconsistent value: Device refused one or more commands:
line vty 5 11
      ^
% Invalid input detected at '^' marker.
Component Response: "
% Modifications of overlapping/sub range is not allowed in controller mode"
Error executing command: CLI command error -
Device(config)# end
```

- You can replace a configuration from an auto consolidation enabled state to an auto consolidation disabled state.

```
Device#show run | sec line
line con 0
stopbits 1
line vty 0 4
transport input all
line vty 5 9
transport input ssh
line vty 10 15
transport input telnet
line vty 16 20
transport input ssh

Device#configure replace bootflash:cfg2.txt
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: y
```

```
Total number of passes: 1
Rollback Done

Device#show run | sec line
no line auto-consolidation
line con 0
stopbits 1
line vty 0 4
transport input all
line vty 5 20
transport input ssh
```

- You can replace a configuration from an auto consolidation disabled state to an auto consolidation enabled state

```
Device#show run | sec line
no line auto-consolidation
line vty 0 4
transport input all
line vty 5 20
transport input ssh

Device#configure replace bootflash:cfg1.txt
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: y
Total number of passes: 1
Rollback Done

Device#show run | sec line
line con 0
stopbits 1
line vty 0 4
transport input all
line vty 5 9
transport input ssh
line vty 10 15
transport input telnet
line vty 16 20
transport input ssh
```

CHAPTER 152

Troubleshooting System Management

Overview

This chapter provides links to documents authored by Cisco subject matter experts (SMEs). They aim to help you resolve technical issues without requiring a support ticket. If these documents are unable to resolve your issue, we recommend visiting the applicable Cisco Community. There is a wealth of information and advice available from fellow Cisco customers who may have experienced this issue already and provided a solution. If you are not able to find a resolution on the Community, it may be best that you raise a support ticket at Cisco Support. In cases where a support ticket has to be raised, these documents provide guidance about the data that should be collected and added to the support ticket. Specify the support document you referred, and TAC can create an improvement request with the document owner.

For information about the licensing packages for features available on the Switches, see the Licensing document.

Feedback Request

Your input helps. A key aspect to improving these support documents is customer feedback. Note that these documents are owned and maintained by multiple teams within Cisco. If you find an issue specific to the document (unclear, confusing, information missing, etc):

- Provide feedback using the **Feedback** button located at the right panel of the corresponding article. The document owner will be notified, and will either update the article, or flag it for removal.
- Include information regarding the section, area, or issue you had with the document and what could be improved. Provide as much detail as possible.

Disclaimer and Caution

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

CHAPTER 153

Dying Gasp

Dying Gasp

The dying gasp feature provides a warning when the power to a network device fails, enabling a temporary power supply. When the host network detects a complete power failure or the removal of a power supply cable, it sends a messsage or signal to the network management system (NMS) through an SNMP trap.

Network devices rely on a temporary backup power supply on a capacitor, which allows for a graceful shutdown and the generation of the dying-gasp message. This temporary power supply is designed to last from 5 to 10 milliseconds to perform these tasks.

Dying gasp packets can be sent to a maximum number of five servers for each notification type.

You configure dying gasp using the following CLI commands:

- **dying-gasp**: Configures the host to create and send dying-gasp packets.
- **show dying-gasp packets**: Displays detailed information about the created packets.
- **snmp-server host**: Specifies the SNMP server for the dying-gasp message.
- **logging host hostname-or-ipaddress transport udp**: Specifies the syslog server sending the syslog dying gasp message.

For further information about dying gasp configuration commands, see the remaining sections of this chapter.

Configuring Dying Gasp

dying-gasp

To enable dying-gasp notification through syslog, SNMP trap, or Ethernet OAM, use the dying-gasp command:

Command Syntax	Description
dying-gasp primary {syslog | snmp-trap | ethernet-oam} secondary {syslog | snmp-trap | ethernet-oam}	• **dying-gasp**: Dying-gasp configuration command • **primary**: Dying-gasp primary notification • **secondary**: Dying-gasp secondary notification • **ethernet-oam**: Enable Ethernet-OAM notification command • **snmp-trap**: Send trap notification to SNMP server command • **syslog**: Enable system logger

The following example shows how to configure SNMP traps as primary notification and syslog as secondary notification:

```
switch(config)# dying-gasp primary snmp-trap secondary syslog
```

show dying-gasp

This section provides descriptions of the **show dying-gasp** command keywords:

Command Syntax	Description
show dying-gasp {status | packets [snmp-trap | syslog | ethernet-oam]}	• **dying-gasp**: Dying-Gasp information • **status**: Dying-Gasp configuration status • **packets**: Detailed information about the created packets • **snmp-trap**: Dying-gasp SNMP trap information • **syslog**: Dying-gasp syslog message information • **ethernet-oam**: Dying-gasp Ethernet OAM message information

show dying-gasp Output Examples

The following text is an example of the **dying gasp packets** command and its output:

```
Switch# show dying-gasp packets
SNMP Trap packet for server 192.168.0.2, link type IP
  interface, via GigabitEthernet1/1, local IP address 12.1.1.40
  encap type is ARPA, local hardware address 6c03.09e7.23c0
  next hop IP address 12.1.1.200, next hop hardware address 6c03.09e7.23c0
Syslog errmsg packet for server 192.168.0.2, link type IP
  interface, via GigabitEthernet1/1, local IP address 12.1.1.40
  encap type is ARPA, local hardware address 6c03.09e7.23c0
  next hop IP address 12.1.1.200, next hop hardware address 6c03.09e7.23c0
```

The following is sample output for the **show dying-gasp status** command:

```
Switch# show dying-gasp status
Dying Gasp Configuration
        SNMP Trap             Enabled (secondary)
        Syslog                Enabled (primary)
```

The following is sample output for the **show dying-gasp packets snmp-trap** command:

```
Switch# show dying-gasp packets snmp-trap
SNMP Trap packet for server 192.168.0.2, link type IP
  interface, via GigabitEthernet1/1, local IP address 12.1.1.40
  encap type is ARPA, local hardware address 6c03.09e7.23c0
  next hop IP address 12.1.1.200, next hop hardware address 6c03.09e7.23c0
```

The following is sample output for the **show dying-gasp packets syslog** command:

```
Switch# show dying-gasp packets syslog
Syslog errmsg packet for server 192.168.0.2, link type IP
  interface, via GigabitEthernet1/1, local IP address 12.1.1.40
  encap type is ARPA, local hardware address 6c03.09e7.23c0
  next hop IP address 12.1.1.200, next hop hardware address 6c03.09e7.23c0
```

CHAPTER 154

Cisco Catalyst Center

Cisco Catalyst Center

The IE3500 devices can be managed and monitored by Catalyst Center, just like other IE switches.

For more details on configuring, managing, and monitoring the IE3500 series devices through Catalyst Center, please refer to the Catalyst Center User Guides available at: https://www.cisco.com/c/en/us/support/cloud-systems-management/dna-center/products-user-guide-list.html.

CHAPTER 155

Configure FPGA Profile

FPGA Profile

The FPGA Profile feature turns certain software features assisted by field-programmable gate array (FPGA) on or off. Some switch features such as Parallel Redundancy Protocol (PRP), High-Availability Seamless Redundancy (HSR), and Device Level Ring (DLR) rely heavily on FPGA implementation. FPGA Profiles allow for efficient allocation of platform resources for the operation of multiple time sensitive, resilient industrial protocols without changes to hardware.

Note FPGA Profiles are supported only on Cisco IE3505 and IE3505H Series Switches. FPGA Profiles are applied globally to the base switch. If an expansion module is present, it uses the same FPGA Profile as configured for the base switch.

The switch supports three FPGA profiles with different combinations of features supported in each profile, as shown in the following table.

FPGA Profile Name	Description
default	Supports 1 instance of PRP or HSR and 3 instance of DLR.
redundancy	Supports 2 instance of PRP or HSR and 2 instance of DLR. **Note** The number of HSR rings supported on the system remains 1 and cannot be increased by the use of expansion module. The same profile that is configured for the switch must be used for the expansion module.

Prerequisites

- Before changing the FPGA Profile, we suggest that you remove the configurations related to the current active FPGA Profile, because those configurations are not valid for the new profile.

Guidelines and Limitations

- FPGA Profile is configured globally on the switch. All base system and expansion modules load the same FPGA Profile configured for the switch. If an expansion module is present, the FPGA Profile configured for the switch also applies to the expansion module.
- You must reload the switch after changing the configured FPGA Profile to activate the profile.
- FPGA Profile is supported in Cisco IOS XE Release 17.17.1.

  To keep the existing profile and feature configurations after an upgrade:

  1. After booting the switch, selected the required FPGA Profile as described in Configure the FPGA Profile, on page 2240.

     Do not copy running-config to startup-config or write memory.
  2. Reload the switch.

     The required feature configurations will not be discarded because they are supported by the selected profile.

Default Settings

The default FPGA Profile name is "default".

Configure the FPGA Profile

Follow these steps to configure the FPGA Profile.

Procedure

Step 1 Use the **fpga-profile activate** command in EXEC mode to activate the required profile:

Example:

```
Switch#fpga-profile activate ?
  default     3 DLR rings and a PRP-channel/HSR-ring
  redundancy  2 DLR rings and 2 PRP-channels/1 HSR-ring
Switch#fpga-profile activate redundancy
```

Step 2 (Optional) Use the **fpga-profile deactivate** command in EXEC mode to deactivate the required profile:

Example:

```
Switch#fpga-profile deactivate
```

Step 3 (Optional) Displays the FPGA Profile that is currently active on the switch:

Example:

```
Switch#show platform fpga-profile ?
  active       current active application profile information
  all          all supported application profile information
  default      default application profile
```

Displays the active FPGA profile and FPGA profile after reload.

```
Switch#sh platform fpga-profile active
Active FPGA profile       : default
Active feature set        : 3 DLR rings and a PRP-channel/HSR-ring

FPGA profile upon reload  : default
Feature set upon reload   : 3 DLR rings and a PRP-channel/HSR-ring
```

Displays default FPGA-profile

```
Switch#sh platform fpga-profile default
FPGA profile  : default
Feature Set   : 3 DLR rings and a PRP-channel/HSR-ring
```

Step 4 Reload the switch:

Example:

```
Switch#reload
```

PART VII

Network Management

CHAPTER 156

Configuring Autoconf

The following sections provide information about Autoconf and how to configure Autoconf:

Prerequisites for Autoconf

- Before enabling Autoconf, disable the Auto SmartPort (ASP) macro, device classifier, and then access the session monitor.

Restrictions for Autoconf

- ASP macro and Autoconf are not supported on the same interface at the same time. Either Autoconf or ASP must be disabled on a per-interface level.
- Interface templates are not applicable for wireless sessions.
- When the Autoconf feature is enabled using the **autoconf enable** command, the default Autoconf service policy is applied to all the interfaces. No other service policy can be applied globally using the **service-policy** command. To apply a different service policy, you must disable Autoconf on that interface. When a service policy is applied globally, you must disable it before enabling the Autoconf feature.
- When both local (interface level) and global service policies exist, the local policy takes precedence. The global service policy comes into effect only when the local policy is removed.
- Service templates cannot be applied to interfaces, and interface templates cannot be applied to service instances.
- Only one service template can be nested inside an interface template.
- Autoconf does not support the switchover feature.

Information about Autoconf

The following sections provide information about Autoconf.

Benefits of Autoconf

The Autoconf feature permits hardbinding between an end device and an interface. Autoconf falls under the umbrella of the Cisco Smart Operations solution. Smart Operations is a comprehensive set of capabilities that can simplify and improve LAN switch deployment, and help organizations deliver operational excellence and scale services on the network.

The Autoconf feature automatically applies the necessary configurations on the device ports to enable the efficient performance of each directly connected end device using a set of interface configurations that are configured inside an interface template:

- Autoconf efficiently applies commands to an interface because the parser does not need to parse each command each time.
- Configurations that are applied through the Autoconf feature can be reliably removed from a port without impacting previous or subsequent configurations on the port.
- The Autoconf feature provides built-in and user-defined configurations using interface and service templates. Configurations applied through templates can be centrally updated with a single operation.
- Using the Autoconf feature, a configuration can be applied to ports and access sessions.
- The Autoconf feature reduces ongoing maintenance for devices and attached end devices by making them intuitive and autoconfigurable. This reduces operation expenses (OPEX) and lowers the total cost of ownership (TCO).

Identity Session Management and Templates

A key advantage of the Autoconf feature is that the core session management capability is decoupled from the application-specific logic, allowing the same framework to be used regardless of the criteria for policy determination or the nature of the policies applied.

The identity session management infrastructure allows configurations or policies or both to be applied as templates.

Both service and interface templates are named as containers of configuration and policy. Service templates can be applied only to access sessions, while interface templates can be applied only to ports. When a service template is applied to an access session, the contained configuration and policy are applied only to the target session, and has no impact on other sessions that may be hosted on the same access port. Similarly, when an interface template is applied to an access port, it impacts all the traffic exchanged on the port.

The Autoconf feature uses a set of built-in maps and built-in templates. The built-in templates are designed based on best practices for interface configurations. Built-in templates can be modified by users to include customized configurations, limiting the need to create a new template.

The templates created by users are referred to as user-defined templates. These templates can be defined on a device and can be mapped to any built-in or user-defined trigger.

Use the **show derived-config** command, to view the overall applied configurations applied by Autoconf template and manual configuration. The interface commands shown in the output of the **show running-config interface** *type number* command are not necessarily the operational configuration. The Autoconf feature dynamically applies a template to the interface, and overrides any conflicting static configuration that is already applied.

Autoconf Operation

Autoconf uses the Device Classifier to identify the end devices that are connected to a port.

The Autoconf feature uses the device classification information gleaned from Cisco Discovery Protocol, LLDP, DHCP, MAC addresses, and the Organizationally Unique Identifier (OUI) that is identified by the Device Classifier.

The Device Classifier provides improved device classification capabilities and accuracy, and increased device visibility for enhanced configuration management.

Device classification is enabled when you enable the Autoconf feature using the **autoconf enable** command in global configuration mode.

The device detection acts as an event trigger, which in turn applies the appropriate automatic template to the interface.

The Autoconf feature is based on a three-tier hierarchy.

- A policy map identifies the trigger type for applying the Autoconf feature.
- A parameter map identifies the appropriate template that must be applied, based on the end device.
- The templates contain the configurations to be applied.

The Autoconf built-in templates and triggers perform the above tasks automatically.

The Autoconf feature provides the following built-in templates:

- AP_INTERFACE_TEMPLATE
- DMP_INTERFACE_TEMPLATE
- IP_CAMERA_INTERFACE_TEMPLATE
- IP_PHONE_INTERFACE_TEMPLATE
- LAP_INTERFACE_TEMPLATE
- MSP_CAMERA_INTERFACE_TEMPLATE
- MSP_VC_INTERFACE_TEMPLATE
- PRINTER_INTERFACE_TEMPLATE
- ROUTER_INTERFACE_TEMPLATE
- SWITCH_INTERFACE_TEMPLATE
- TP_INTERFACE_TEMPLATE

Note By default, built-in templates are not displayed under running configuration. The built-in templates are displayed in the running configuration only if you edit them.

The template that is selected is based on parameter map information applied to an interface. This information can be based on the following criteria:

- End Device type
- MAC address
- OUI
- Platform type
- User role
- Username

The Autoconf feature provides one built-in parameter map (BUILTIN_DEVICE_TO_TEMPLATE) with the following configuration:

```
Parameter-map name: BUILTIN_DEVICE_TO_TEMPLATE
 Map: 10 map device-type regex "Cisco-IP-Phone"
  Action(s):
   20 interface-template IP_PHONE_INTERFACE_TEMPLATE
 Map: 20 map device-type regex "Cisco-IP-Camera"
  Action(s):
   20 interface-template IP_CAMERA_INTERFACE_TEMPLATE
 Map: 30 map device-type regex "Cisco-DMP"
  Action(s):
   20 interface-template DMP_INTERFACE_TEMPLATE
 Map: 40 map oui eq "00.0f.44"
  Action(s):
   20 interface-template DMP_INTERFACE_TEMPLATE
 Map: 50 map oui eq "00.23.ac"
  Action(s):
   20 interface-template DMP_INTERFACE_TEMPLATE
 Map: 60 map device-type regex "Cisco-AIR-AP"
  Action(s):
   20 interface-template AP_INTERFACE_TEMPLATE
 Map: 70 map device-type regex "Cisco-AIR-LAP"
  Action(s):
   20 interface-template LAP_INTERFACE_TEMPLATE
 Map: 80 map device-type regex "Cisco-TelePresence"
  Action(s):
   20 interface-template TP_INTERFACE_TEMPLATE
 Map: 90 map device-type regex "Surveillance-Camera"
  Action(s):
   10 interface-template MSP_CAMERA_INTERFACE_TEMPLATE
 Map: 100 map device-type regex "Video-Conference"
  Action(s):
   10 interface-template MSP_VC_INTERFACE_TEMPLATE
```

Note Use the **show parameter-map type subscriber attribute-to-service All** command to view the configuration for the built-in parameter map.

The Autoconf feature provides one built-in policy map (BUILTIN_AUTOCONF_POLICY) with the following configuration:

```
BUILTIN_AUTOCONF_POLICY
  event identity-update match-all
    10 class always do-until-failure
      10 map attribute-to-service table BUILTIN_DEVICE_TO_TEMPLATE
```

Note Use the **show policy-map type control subscriber BUILTIN_AUTOCONF_POLICY** command to view the configuration for the built-in policy map.

You can also manually create policy maps, parameter maps, and templates.

When a trigger is created that is based on specific user information, a local 802.1X Cisco Identity Services Engine (ISE) server authenticates it, ensuring the security of the operation.

An interface template can be dynamically activated (on an interface) using any of the following methods:

- RADIUS CoA: While Change of Authorization (CoA) commands are targeted at one or more access sessions, any referenced template must be applied to the interface that is hosting the referenced session.
- RADIUS Access-Accept for client authentication or authorization: Any referenced interface template returned in an Access-Accept must be applied to the port that is hosting the authorized access session.
- Service template: If an interface template is referenced in a service template that is either locally defined or sourced from the AAA server, the interface template must be applied to the interface hosting an access-session on which the service template is applied. (Add a new command for interface template reference from within a locally defined service template.)
- Subscriber control-policy action: A mapping action under the subscriber control policy activates service or interface template (as referenced in a parameter map) or both based on the type of filter, and removes templates, if any, associated with a previous policy.
- Device-to-template parameter map: A subscriber parameter map that allows the filter type-to-service or interface template mappings or both to be specified in an efficient and readable manner.

Advantages of Using Templates

Using templates for auto configuration has the following benefits:

- Templates are parsed once when they are being defined. This makes the dynamic application of the templates very efficient.
- Templates can be applied to an Ethernet interface that is connected to an end device, based on the type of end device.
- Service templates allow the activation of session-oriented features, whereas interface templates apply configurations to the interface that is hosting a session.
- Service templates are applied to access sessions and hence only impact the traffic exchanged with a single endpoint on a port.
- Startup and running configurations of a device are not modified by the dynamic application of a template.
- Policy application is synchronized with the access-session life cycle. This is tracked by the framework by using all the available techniques, including link-up or link-down.

- Templates can be updated with a single operation. All the applied instances of templates are also updated during this operation.
- Constituent commands of templates do not appear in the running configuration.
- Templates can be removed with no impact on previous or subsequent configurations.
- Template application is acknowledged, allowing for synchronization and performance of remedial actions when failures occur.
- Data VLAN, quality of service (QoS) parameters, storm control, and MAC-based port security are configured automatically based on the end device that is connected to the switch.
- The switch port is cleaned up completely by removing configurations when the device is disconnected from a port.
- Human error is reduced in the installation and configuration process.

Autoconf Functionality

The Autoconf feature is disabled by default in global configuration mode. When you enable the Autoconf feature in global configuration mode, it is enabled by default at the interface level. The built-in template configurations are applied based on the end devices detected on all the interfaces.

Use the **access-session inherit disable autoconf** command to manually disable Autoconf at the interface level, even when Autoconf is enabled at the global level.

If you disable Autoconf at the global level, all the interface-level configurations are disabled.

Table 158: Autoconf Functionality

Global	Interface Level	AutoConf Status
Disable	Disable	No automatic configurations are applied when an end device is connected.
Enable	Enable	If Autoconf is enabled at the global level, it is also enabled at the interface level by default. Built-in template configurations are applied based on the end devices that are detected on all the interfaces.
Enable	Disable	Enabled at global level. Disabled at interface level. No automatic configurations are applied when an end device is connected to the interface on which Autoconf is disabled.

Autoconf allows you to retain the template even when the link to the end device is down or the end device is disconnected, by configuring the autoconf sticky feature **access-session interface-template sticky** command in global configuration mode. The Autoconf sticky feature avoids the need for detecting the end device and applying the template every time the link flaps or the device is removed and connected back.

The **access-session interface-template sticky** command is mandatory to apply an inbuilt template that contains **access-session** commands on an interface. Configure the **access-session interface-template sticky** command to apply interface template on a port using a service policy.

To disable the Autoconf feature on a specific interface, use the **access-session inherit disable interface-template-sticky** command in interface configuration mode.

How to Configure Autoconf

The following sections provide information about how to configure Autoconf.

Applying a Built-In Template to an End Device

The following task shows how to apply a built-in template on an interface that is connected to an end device, for example, a Cisco IP phone.

Before you begin

Make sure that the end device, for example, a Cisco IP phone, is connected to a switch port.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**autoconf enable** **Example:** `Device(config)# autoconf enable`	Enables the Autoconf feature.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.
Step 5	**show device classifier attached interface** *interface-type interface-number* **Example:** `Device# show device classifier attached interface Gi3/0/26`	(Optional) Displays whether the end device is classified by the device classifier with correct attributes.
Step 6	**show template binding target** *interface-type interface-number* **Example:** `Device# show template binding target gi3/0/26`	Displays the configuration applied through the template on the interface.

Example

The following example shows that an IP phone is classified by the device classifier with correct attributes:

```
Device# show device classifier attached interface GigabitEthernet 1/1

Summary:

MAC_Address       Port_Id       Profile Name                      Device Name
==============    ==========    ==================                ==================
0026.0bd9.7bbb    Gi3/0/26      Cisco-IP-Phone-7962               Cisco IP Phone 7962
```

The following example shows that a built-in interface template is applied on an interface:

```
Device# show template binding target GigabitEthernet 1/1

Interface Templates
===================
Interface: Gi4/0/11
Method              Source            Template-Name
------              ------            -------------
dynamic             Built-in          IP_PHONE_INTERFACE_TEMPLATE
```

The following example shows how to verify the interface configuration after the interface template is applied to an IP phone connected to the GigabitEthernet interface 1/1:

```
Device# show running-config interface GigabitEthernet 1/1
Building configuration...

Current configuration : 624 bytes
!
interface GigabitEthernet1/1
!
End

Device# show derived-config interface GigabitEthernet1/1

Building configuration...

Derived configuration : 649 bytes
!
interface GigabitEthernet1/1
 switchport mode access
 switchport block unicast
 switchport port-security maximum 3
 switchport port-security maximum 2 vlan access
 switchport port-security violation restrict
 switchport port-security aging time 2
 switchport port-security aging type inactivity
 switchport port-security
 load-interval 30
 storm-control broadcast level pps 1k
 storm-control multicast level pps 2k
 storm-control action trap
 spanning-tree portfast
 spanning-tree bpduguard enable
 service-policy input AutoConf-4.0-CiscoPhone-Input-Policy
 service-policy output AutoConf-4.0-Output-Policy
 ip dhcp snooping limit rate 15
```

```
end
```

The following example shows how to verify the global configuration after configuring Autoconf:

```
Device# show running config
class-map match-any AutoConf-4.0-Scavenger-Queue
 match dscp cs1
 match cos  1
 match access-group name AutoConf-4.0-ACL-Scavenger
class-map match-any AutoConf-4.0-VoIP
 match dscp ef
 match cos  5
class-map match-any AutoConf-4.0-Control-Mgmt-Queue
 match cos  3
 match dscp cs7
 match dscp cs6
 match dscp cs3
 match dscp cs2
 match access-group name AutoConf-4.0-ACL-Signaling
class-map match-any AutoConf-4.0-Multimedia-Conf
 match dscp af41
 match dscp af42
 match dscp af43
class-map match-all AutoConf-4.0-Broadcast-Vid
 match dscp cs5
class-map match-any AutoConf-4.0-Bulk-Data
 match dscp af11
 match dscp af12
 match dscp af13
class-map match-all AutoConf-4.0-Realtime-Interact
 match dscp cs4
class-map match-any AutoConf-4.0-VoIP-Signal
 match dscp cs3
 match cos  3
class-map match-any AutoConf-4.0-Trans-Data-Queue
 match cos  2
 match dscp af21
 match dscp af22
 match dscp af23
 match access-group name AutoConf-4.0-ACL-Transactional-Data
class-map match-any AutoConf-4.0-VoIP-Data
 match dscp ef
 match cos  5
class-map match-any AutoConf-4.0-Multimedia-Stream
 match dscp af31
 match dscp af32
 match dscp af33
class-map match-all AutoConf-4.0-Internetwork-Ctrl
 match dscp cs6
class-map match-all AutoConf-4.0-VoIP-Signal-Cos
 match cos  3
class-map match-any AutoConf-4.0-Multimedia-Stream-Queue
 match dscp af31
 match dscp af32
 match dscp af33
class-map match-all AutoConf-4.0-Network-Mgmt
 match dscp cs2
class-map match-all AutoConf-4.0-VoIP-Data-Cos
 match cos  5
class-map match-any AutoConf-4.0-Priority-Queue
 match cos  5
 match dscp ef
 match dscp cs5
 match dscp cs4
```

```
class-map match-any AutoConf-4.0-Bulk-Data-Queue
 match cos  1
 match dscp af11
 match dscp af12
 match dscp af13
 match access-group name AutoConf-4.0-ACL-Bulk-Data
class-map match-any AutoConf-4.0-Transaction-Data
 match dscp af21
 match dscp af22
 match dscp af23
class-map match-any AutoConf-4.0-Multimedia-Conf-Queue
 match cos  4
 match dscp af41
 match dscp af42
 match dscp af43
 match access-group name AutoConf-4.0-ACL-Multimedia-Conf
class-map match-all AutoConf-4.0-Network-Ctrl
 match dscp cs7
class-map match-all AutoConf-4.0-Scavenger
 match dscp cs1
class-map match-any AutoConf-4.0-Signaling
 match dscp cs3
 match cos  3
!
!
policy-map AutoConf-4.0-Cisco-Phone-Input-Policy
 class AutoConf-4.0-VoIP-Data-Cos
  set dscp ef
  police cir 128000 bc 8000
   exceed-action set-dscp-transmit cs1
   exceed-action set-cos-transmit 1
 class AutoConf-4.0-VoIP-Signal-Cos
  set dscp cs3
  police cir 32000 bc 8000
   exceed-action set-dscp-transmit cs1
   exceed-action set-cos-transmit 1
 class class-default
  set dscp default
  set cos 0
policy-map AutoConf-4.0-Output-Policy
 class AutoConf-4.0-Scavenger-Queue
  bandwidth remaining percent 1
 class AutoConf-4.0-Priority-Queue
  priority
  police cir percent 30 bc 33 ms
 class AutoConf-4.0-Control-Mgmt-Queue
  bandwidth remaining percent 10
 class AutoConf-4.0-Multimedia-Conf-Queue
  bandwidth remaining percent 10
 class AutoConf-4.0-Multimedia-Stream-Queue
  bandwidth remaining percent 10
 class AutoConf-4.0-Trans-Data-Queue
  bandwidth remaining percent 10
    dbl
 class AutoConf-4.0-Bulk-Data-Queue
  bandwidth remaining percent 4
    dbl
 class class-default
  bandwidth remaining percent 25
    dbl
policy-map AutoConf-DMP
 class class-default
  set dscp cs2
policy-map AutoConf-IPVSC
```

```
 class class-default
  set cos dscp table AutoConf-DscpToCos
policy-map AutoConf-4.0-Input-Policy
 class AutoConf-4.0-VoIP
 class AutoConf-4.0-Broadcast-Vid
 class AutoConf-4.0-Realtime-Interact
 class AutoConf-4.0-Network-Ctrl
 class AutoConf-4.0-Internetwork-Ctrl
 class AutoConf-4.0-Signaling
 class AutoConf-4.0-Network-Mgmt
 class AutoConf-4.0-Multimedia-Conf
 class AutoConf-4.0-Multimedia-Stream
 class AutoConf-4.0-Transaction-Data
 class AutoConf-4.0-Bulk-Data
 class AutoConf-4.0-Scavenger
```

Applying a Modified Built-In Template to an End Device

The following task shows how to modify a built-in template when multiple wireless access points and IP cameras are connected to a switch:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**template** *template-name* **Example:** `Device(config)# template AP_INTERFACE_TEMPLATE`	Enters template configuration mode for the built-in template.
Step 4	**switchport access vlan** *vlan-id* **Example:** `Device(config-template)# switchport access vlan 20`	Sets the VLAN when the interface is in access mode.
Step 5	**description** *description* **Example:** `Device(config-template)# description modifiedAP`	Modifies the description of the built-in template.
Step 6	**exit** **Example:**	Exits template configuration mode and enters global configuration mode.

	Command or Action	Purpose
	`Device(config-template)# exit`	
Step 7	**autoconf enable** **Example:** `Device(config)# autoconf enable`	Enables the Autoconf feature.
Step 8	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.
Step 9	**show template interface binding all** **Example:** `Device# show template interface binding all`	Displays whether the template is applied on the interface.

Example

The following example shows that an IP camera and access points are classified by the device classifier with correct attributes:

```
Device# show device classifier attached detail

DC default profile file version supported = 1

Detail:
MAC_Address     Port_Id    Cert Parent Proto    ProfileType     Profile Name
Device_Name
=============================================================================================================
001d.a1ef.23a8  Gi1/1     30    3    C     M    Default      Cisco-AIR-AP-1130      cisco
AIR-AP1131AG-A-K9
001e.7a26.eb05  Gi1/1     70    2    C     M    Default      Cisco-IP-Camera        Cisco
IP Camera
```

The following example shows that the built-in interface template is applied on an interface:

```
Device# show template interface binding all

Template-Name                   Source            Method          Interface
-------------                   ------            ------          ---------
IP_CAMERA_INTERFACE_TEMPLATE     Built-in          dynamic         Gi1/1
AP_INTERFACE_TEMPLATE          Modified-Built-in   dynamic         Gi1/1
```

Migrating from ASP to Autoconf

Before you begin

Verify that the AutoSmart Port (ASP) macro is running by using the **show running-config | include macro auto global** command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no macro auto global processing** **Example:** `Device(config)# no macro auto global processing`	Disables ASP on a global level.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**clear macro auto configuration all** **Example:** `Device# clear macro auto configuration all`	Clears macro configurations for all interfaces.
Step 6	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 7	**autoconf enable** **Example:** `Device(config)# autoconf enable`	Enables the Autoconf feature.
Step 8	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring a Platform Type Filter

The following tasks shows how to configure a platform type filter for class maps and parameter maps.

Configuring a Platform Type Filter for a Class Map

A control class defines the conditions under which the actions of a control policy are executed. You should define whether all, any, or none of the conditions must be evaluated to execute the actions of the control policy. Platform types are evaluated based on the specified platform name in the control policy.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map type control subscriber** {**match-all** | **match-any** | **match-none**} *control-class-name* **Example:** `Device(config)# class-map type control subscriber match-all DOT1X_NO_AGENT`	Creates a control class and enters control class-map filter mode. • **match-all**: Must match all the conditions in the control class. • **match-any**: Must match at least one condition in the control class. • **match-none**: Must not match any of the conditions in the control class.
Step 4	**match platform-type** *platform-name* **Example:** `Device(config-filter-control-classmap)# match platform-type C3850`	Creates a condition to evaluate control classes based on the specified platform type.
Step 5	**end** **Example:** `Device(config-filter-control-classmap)# end`	Exits control class-map filter mode and returns to privileged EXEC mode.
Step 6	**show class-map type control subscriber** {**all** | **name** *control-class-name*} **Example:** `Device# show class-map type control subscriber all`	(Optional) Displays information about control policies for all the class maps or a specific class map.

Configuring a Platform Type Filter for a Parameter Map

We recommend that you use the parameter map.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**parameter-map type subscriber attribute-to-service** *parameter-map-name* **Example:** `Device(config)# parameter-map type subscriber attribute-to-service Aironet-Policy-para`	Specifies the parameter map type and name, and enters parameter-map filter mode.
Step 4	*map-index* **map platform-type** {{**eq** | **not-eq** | **regex**} *filter-name*} **Example:** `Device(config-parameter-map-filter)# 10 map platform-type eq C9xxx`	Specifies the parameter map attribute filter criteria to the platform type.
Step 5	**end** **Example:** `Device(config-parameter-map-filter-submode)# end`	Exits parameter-map filter mode and returns to privileged EXEC mode.
Step 6	**show parameter-map type subscriber attribute-to-service**{**all | name** *parameter-map-name*} **Example:** `Device# show parameter-map type subscriber attribute-to-service`	(Optional) Displays the parameter map attributes.

Configuring a Device Type Filter for a Class Map

A control class defines the conditions under which the actions of a control policy are executed. You should define whether all, any, or none of the conditions must be evaluated to execute the actions of the control policy. Device types are evaluated based on the specified device name in the control policy.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map type control subscriber** {**match-all** | **match-any** | **match-none**} *control-class-name* **Example:** `Device(config)# class-map type control subscriber match-all Device-Type-Match`	Creates a control class and enters control class-map filter mode. • **match-all**: Must match all the conditions in the control class. • **match-any**: Must match at least one condition in the control class. • **match-none**: Must not match any of the conditions in the control class.
Step 4	**match device-type** {*device-name*|**regex** *regular-expression*} **Example:** `Device(config-filter-control-classmap)# match device-type laptop` `Device(config-filter-control-classmap)# match device-type regex cis*`	Creates a condition to evaluate control classes based on the specified device type. • *device-name*: Enter a device name for the class map attribute filter criteria. • **regex** *regular-expression*: Enter a regular expression to specify the filter type.
Step 5	**end** **Example:** `Device(config-filter-control-classmap)# end`	Exits control class-map filter mode and returns to privileged EXEC mode.
Step 6	**show class-map type control subscriber** {**all** | **name** *control-class-name*} **Example:** `Device# show class-map type control subscriber all`	(Optional) Displays information about control policies for all the class maps or a specific class map.

Configuration Examples for Autoconf

The following sections provide configuration examples for the Autoconf feature.

Example: Applying a Built-In Template to an End Device

The following example shows how to apply a built-in template to an end device connected to an interface:

```
Device> enable
Device(config)# configure terminal
Device(config)# autoconf enable
Device(config)# end
Device# show device classifier attached interface Gi3/0/26
Device# show template binding target GigabitEthernet 1/1
```

Example: Applying a Modified Built-In Template to an End Device

The following example shows how to apply a modified built-in template to an end device and verify the configuration:

```
Device> enable
Device(config)# configure terminal
Device(config)# template AP_INTERFACE_TEMPLATE
Device(config-template)# switchport access vlan 20
Device(config-template)# description modifiedAP
Device(config-template)# exit
Device(config)# autoconf enable
Device(config)# end
Device# show template interface binding all
```

Example: Migrating from ASP Macros to Autoconf

The following example shows how to migrate from ASP to Autoconf:

```
Device> enable
Device# configure terminal
Device(config)# no macro auto global processing
Device(config)# exit
Device# clear macro auto configuration all
Device# configure terminal
Device(config)# autoconf enable
Device(config)# end
```

Example: Configuring a Platform Type Filter

The following example shows how to configure a platform type filter for a class map:

```
Device> enable
Device# configure terminal
Device(config)# class-map type control subscriber match-all DOT1X_NO_AGENT
Device(config-filter-control-classmap)# match platform-type C9xxx
Device(config-filter-control-classmap)# end
Device#
```

The following example shows how to configure a platform type filter for a parameter map:

```
Device> enable
Device# configure terminal
```

```
Device(config)# parameter-map type subscriber attribute-to-service Aironet-Policy-para
Device(config-parameter-map-filter)# 10 map platform-type eq C9xxx
Device(config-parameter-map-filter-submode)# end
Device#
```

CHAPTER 157

Configuring Interface Templates

The following sections provide information about Interface Templates and how to configure and bind Interface Templates to a target:

Restrictions for Interface Templates

- Remote storing and downloading of templates is not supported.
- To dynamically bind an interface template, the interface template with the same name as referred by AAA Authorization has to be configured on the device.

Information About Interface Templates

This section describes an interface template, its types and usage.

Interface Template Overview

An interface template is a container of configurations or policies that can be applied to specific ports. When an interface template is applied to an access port, it impacts all traffic that is exchanged on the port.

There are two types of interface templates; user and builtin templates. Builtin templates are created by the system.

You can modify builtin templates. If you delete a modified builtin template, the system restores the original definition of the template.

The following are the available builtin templates:

- AP_INTERFACE_TEMPLATE (Access Point)
- DMP_INTERFACE_TEMPLATE (Digital Media Player)
- IP_CAMERA_INTERFACE_TEMPLATE

- IP_PHONE_INTERFACE_TEMPLATE
- LAP_INTERFACE_TEMPLATE (Lightweight Access Point)
- MSP_CAMERA_INTERFACE_TEMPLATE
- MSP_VC_INTERFACE_TEMPLATE (Video Conferencing)
- PRINTER_INTERFACE_TEMPLATE
- ROUTER_INTERFACE_TEMPLATE
- SWITCH_INTERFACE_TEMPLATE
- TP_INTERFACE_TEMPLATE (TelePresence)

Following is an example of a builtin interface template:

```
Template Name       : IP_CAMERA_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
```

You can also create specific user templates with the commands that you want to include.

Note The template name must not contain spaces.

You can create an interface template using the **template** command in global configuration mode. In template configuration mode, enter the required commands. The following commands can be entered in template configuration mode:

Command	Description
access-session	Configures access session-specific interface commands. This is applicable to Identity-Based Networking Services (IBNS) 2.0
authentication	Configures authentication manager Interface Configuration commands. This is applicable to IBNS1.0
carrier-delay	Configures delay for interface transitions.
cts manual	Supplies local configuration for Cisco TrustSec (CTS) parameters and puts the device in CTS manual interface configuration mode.
default	Sets a command to its defaults.

Command	Description
description	Configures interface-specific description.
dot1x	Configures interface configuration commands for IEEE 802.1X.
ip	Configures IP template.
keepalive	Enables keepalive.
load-interval	Specifies interval for load calculation for an interface.
mab	Configures MAC authentication bypass Interface.
peer	Configures peer parameters for point to point interfaces.
service-policy	Configures CPL service policy.
source	Gets configurations from another source.
spanning-tree	Configures spanning tree subsystem
storm-control	Configures storm control.
subscriber	Configures subscriber inactivity timeout value.
switchport	Sets switching mode configurations

Note System builtin templates aren't displayed in the running configuration. These templates show up in the running configuration only if you edit them.

Binding an Interface Template to a Target

Each template can be bound to a target, like an interface or a sub-interface. A template can be attached to a target either statically or dynamically. Static binding of a template involves binding the template to a target, like an interface. Only one template can be bound at a time using static binding. Static binding of another template to the same target will unbind the previously bound template. To configure static binding, use the **source template** command in interface configuration mode.

Any number of templates can be bound dynamically to a target. To configure dynamic binding using builtin policy maps and parameter maps, enable the Autoconf feature using the **autoconf enable** command.

Note You can have statically and dynamically bound templates on the same interface at a time.

Priority for Configurations Using Interface Templates

Configuration applied through dynamically-bound templates has the highest priority, followed by configuration applied directly on the interface, and then configuration applied through statically-bound templates. When similar commands are present at different priority levels, the one at the highest priority is applied. If a configuration at a higher priority level is not applied, then the configuration with the next highest priority is applied to the target.

Multiple templates can be dynamically bound to a target. When multiple templates are dynamically bound, the template that is applied last has the highest priority.

To delete a template, you must remove the binding to all targets. If you bind a template that does not exist, a new template is created with no configurations.

How to Configure Interface Templates

Perform the following tasks to configure a user interface template and bind it to a target.

Configuring Interface Templates

Perform the following task to create user interface templates:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**template** *name* **Example:** `Device(config)# template user-template1`	Creates a user template and enters template configuration mode. **Note** Builtin templates are system-generated.
Step 4	**load-interval** *interval* **Example:** `Device(config-template)# load-interval 60`	Configures the sampling interval for statistics collections on the template. **Note** Builtin templates are system-generated.
Step 5	**description** *description* **Example:**	Configures the description for the template.

	Command or Action	Purpose
	`Device(config-template)# description This is a user template`	
Step 6	**keepalive** *number* **Example:** `Device(config-template)# Keepalive 60`	Configures the keepalive timer.
Step 7	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Static Binding for Interface Templates

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies the interface type and number and enters interface configuration mode.
Step 4	**source template** *name* **Example:** `Device(config-if)# source template user-template1`	Statically applies an interface template to a target.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example

To verify static binding use the **show running-config interface** *int-name* and the **show derived-config interface** *int-name* commands.

```
Device# show running-config interface GigabitEthernet 1/1

Building configuration...

Current configuration : 71 bytes
!
interface GigabitEthernet1/1
source template user-template1
end
```

```
Device# show derived-config interface GigabitEthernet 1/1
Building configuration...

Derived configuration : 108 bytes
!
interface GigabitEthernet1/1
description This is a user template
load-interval 60
keepalive 60
end
```

Configuring Dynamic Binding of Interface Templates

To configure dynamic binding of interface templates, perform the following tasks:

Before you begin

Ensure that 802.1x port-based authentication is configured on the device. When you are using the ISE server to download and assign a template that includes **switchport mode** and vlan change commands, the **access-session interface-template sticky** command is used, which is available only in IBNS 2.0. Hence, VLAN changes using a template require IBNS 2.0.

Procedure

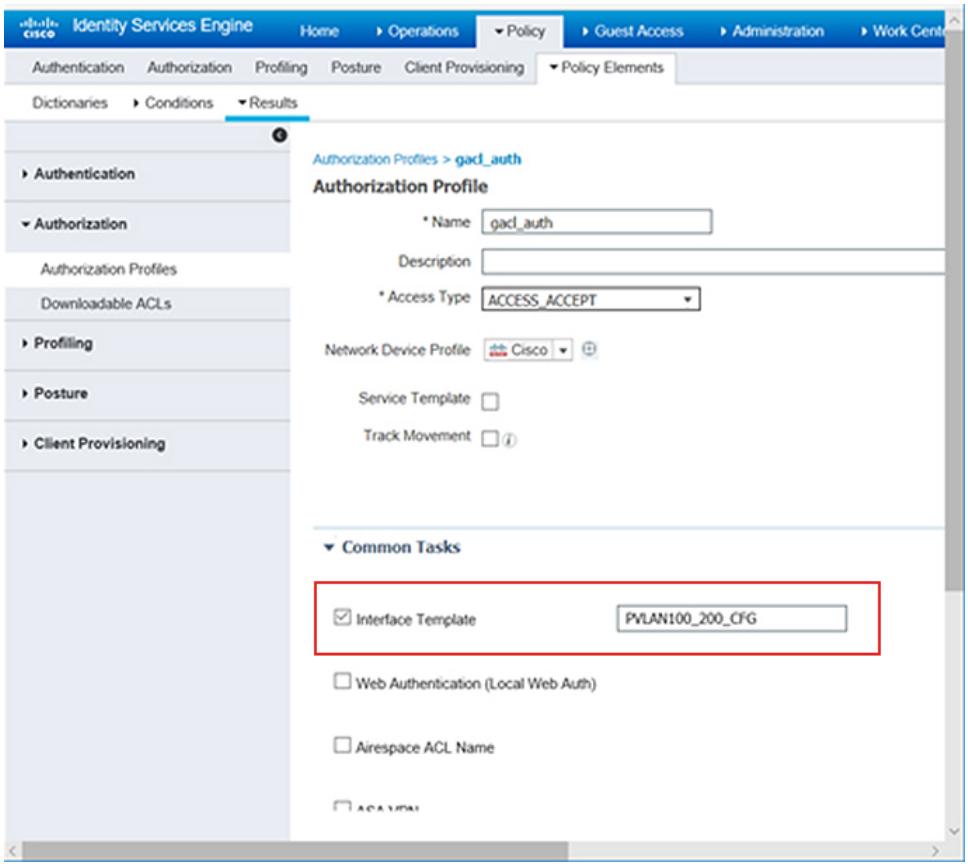
Step 1 Create a template on the device, as specified in the Configuring Interface Templates, on page 2266 procedure.

Step 2 Configure the Identity Services Engine (ISE) or any other RADIUS server to download the template name to the device interface. The template is assigned after the device is authenticated successfully.

If you're using ISE, go to the **Policy** > **Policy Elements** > **Authorization** > **Authorization Profile** page.

Check the **Interface Template** check box and enter the name of the template to be assigned to the client interface.

Figure 144: Configure ISE to Assign Interface Template



If you're using a different RADIUS server, configure the attribute **Cisco-AVpair="interface:template=*name*"** with the name of the template. This configuration pushes the template to the device after the initial client authentication is completed.

Step 3 To verify that the template name is downloaded to the interface, use the **show access-session interface** *interface-id* **details** or **show access-session interface** *interface-id* **details** command. To verify that the interface template commands are applied to the interface, use the **show derived-config interface** *interface-id* command.

Example

The following example shows how to verify that a template named del_template is downloaded and applied to the gigabitethernet 1/1 interface on the device:

```
Device# show running-config | section del_template
template del_template
access-session port-control auto
no access-session monitor
authentication periodic

Device# show access-session interface gigabitethernet 1/1 details
Interface: gigabitethernet 1/1
IIF-ID: 0x1F9EBBA9
MAC Address: 002f.0100.0001
```

```
IPv6 Address: Unknown
IPv4 Address: Unknown
User-Name: NOAS
Device-type: Un-Classified Device
Device-name: Unknown Device
Status: Authorized
Domain: DATA
Oper host mode: multi-auth
Oper control dir: both
Session timeout: N/A
Common Session ID: 0404140B00004E1C6E569E0B
Acct Session ID: Unknown
Handle: 0xdb000e24
Current Policy: DOT1x

Server Policies:
Interface Template: del_template

Method status list:
Method State
dot1x Authc Success


Device# show derived-config interface gigabitethernet 1/1
Building configuration...

Derived configuration : 321 bytes
!
interface gigabitethernet 1/1
switchport access vlan 44
switchport mode access
switchport port-security violation restrict
switchport port-security
authentication periodic
access-session port-control auto
no access-session monitor
mab
dot1x pae authenticator
service-policy type control subscriber DOT1x
end
```

Verifying Interface Templates

Use one or more of the commands listed below to verify the interface template configuration.

Table 159: Show commands to verify Interface Template Configuration

Command	Purpose
show template interface all {**all** | **binding** {*temp-name* | **all** | **target** *int-name*} | **brief** }	Shows all interface template configurations.
show template interface source {**built-in** [**original**] | **user**}{*temp-name* | **all**}}	Shows interface template source configurations.
show template service{**all** | **binding target** *int-name* | **brief** | **source** {**aaa** | **built-in** | **user** {**temp-name** | **all**}}	Shows all interface template service configurations.

Example: Verifying Interface User Template

```
Device# show template interface source user all
    Template Name : TEST-1
    Template Definition:
    load-interval 60
    description TEST_1_TEMPLATE
    keepalive 200
    !
    Template Name : TEST-2
    Template Definition:
    load-interval 60
    description TEST-1_TEMPLATE
    keepalive 200
```

Example: Verifying all Builtin Templates

```
Device#  show template interface source built-in all

Building configuration...

Template Name : AP_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode trunk
switchport nonegotiate
service-policy input AutoConf-4.0-Trust-Cos-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
!
Template Name : DMP_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport block unicast
switchport port-security
spanning-tree portfast
spanning-tree bpduguard enable
service-policy input AutoConf-4.0-Trust-Dscp-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
!
Template Name : IP_CAMERA_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport block unicast
switchport port-security
spanning-tree portfast
spanning-tree bpduguard enable
service-policy input AutoConf-4.0-Trust-Dscp-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
!
Template Name : IP_PHONE_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport block unicast
switchport port-security maximum 3
switchport port-security maximum 2 vlan access
switchport port-security violation restrict
switchport port-security aging time 2
switchport port-security aging type inactivity
switchport port-security
storm-control broadcast level pps 1k
storm-control multicast level pps 2k
```

```
storm-control action trap
spanning-tree portfast
spanning-tree bpduguard enable
service-policy input AutoConf-4.0-CiscoPhone-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
ip dhcp snooping limit rate 15
load-interval 30
!
Template Name : LAP_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport block unicast
switchport port-security violation protect
switchport port-security aging time 2
switchport port-security aging type inactivity
switchport port-security
storm-control broadcast level pps 1k
storm-control multicast level pps 2k
storm-control action trap
spanning-tree portfast
spanning-tree bpduguard enable
ip dhcp snooping limit rate 15
load-interval 30
!
Template Name : MSP_CAMERA_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport block unicast
switchport port-security
spanning-tree portfast
spanning-tree bpduguard enable
!
Template Name : MSP_VC_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport port-security
spanning-tree portfast
spanning-tree bpduguard enable
load-interval 30
!
Template Name : PRINTER_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport port-security maximum 2
switchport port-security
spanning-tree portfast
spanning-tree bpduguard enable
load-interval 60
!
Template Name : ROUTER_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode trunk
spanning-tree portfast trunk
spanning-tree bpduguard enable
service-policy input AutoConf-4.0-Trust-Cos-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
!
Template Name : SWITCH_INTERFACE_TEMPLATE
Modified : No
```

```
Template Definition :
switchport mode trunk
service-policy input AutoConf-4.0-Trust-Cos-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
!
Template Name : TP_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport port-security maximum 3
switchport port-security maximum 2 vlan access
switchport port-security violation restrict
switchport port-security aging time 2
switchport port-security aging type inactivity
switchport port-security
storm-control broadcast level pps 1k
storm-control multicast level pps 2k
storm-control action trap
spanning-tree portfast
spanning-tree bpduguard enable
service-policy input AutoConf-4.0-Trust-Dscp-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
ip dhcp snooping limit rate 15
load-interval 30
!
end
```

Example: Verifying Interface Template Binding for all templates

```
Device# show template interface binding all
    Template-name               Source      Method          Interface
    =============               ====      ==========      ==========
IP_PHONE_INTERFACE_TEMPLATE    Built-in    Dynamic        Gi1/1


IP_PHONE_INTERFACE_TEMPLATE    Built-in     Static        Gi1/1
```

Example: Verifying Static Template Binding for a Target Interface

```
Device# show template interface binding target GigabitEthernet 1/1
    Interface            Method       Source            Template
    =========           ==========    =====             =========
    Gi1/1              Dynamic      built-in          IP_PHONE_INTERFACE_TEMPLATE
                         Static       user              TEST
                         Dynamic      Modified-built-in TEST
```

Example: Verifying Dynamic Template Binding for all templates

```
Device# show template service all

    User-defined template:
    ======================

    Template Name      : SVC-1
    Template Definition:
    vlan 100
    access-group acl1

    built-in template:
    ======================

    Template Name      : SVC-2
    Template Definition:
```

```
vlan 100
access-group acl1

aaa downloaded template:
==========================
Template Name      : SVC-2
Template Definition:
vlan 100
access-group acl1
```

Example: Verifying Template Binding for a Target Interface

```
Device# show template binding target GigabitEthernet 1/1

  Interface Templates:
    Interface            method       Source            Template
    =========           ==========    =====             =========
    Gi1/1               Dynamic      built-in          IP_PHONE_INTERFACE_TEMPLATE
                          Static       user              TEST
                          Dynamic      Modified-built-in TEST
  Service Templates:
   Template             Source         Session-Mac
   ========             ====           ================
   SVC1                 user           aa-bb-cc-dd-ee-ff
   SVC2                 built-in       ab-ab-ab-ab-ab-ab
   SVC3                 aaa            ac-ac-ac-ac-ac-ac
```

Configuration Examples for Interface Templates

Example: Configuring User Interface Templates

Example: Configuring User Templates

```
Device# enable
Device (config)# configure terminal
Device(config)# template user-template1
Device(config-template)# load-interval 60
Device(config-template)# description This is a user template
Device(config-template)# Keepalive 60
Device(config)# end
```

Example: Sourcing Interface Templates

```
Device> enable
Device# configure terminal
Device(config-if)# source template user-template1
Device(config-if)# end
```

Example: Dynamically Binding Interface Templates

Configure a template on the device:

```
Device# configure terminal
Device(config)# tempalte user_template
Device(config-template)# access-session port-control auto
Device(config-template)# no access-session monitor
Device(config-template)# authentication periodic
```

Configure RADIUS Server attribute with the template name:

```
Cisco-AVpair="interface:template=user_template"
```

CHAPTER 158

Configuring Cisco Plug and Play

Configuring Cisco Plug and Play

For information about configuring Plug and Play, see

- Cisco Plug and Play Feature Guide
- Configuration Guide for Cisco Network Plug and Play on APIC-EM

CHAPTER 159

Configuring Cisco Discovery Protocol

Cisco Discovery Protocol is a Layer 2, media-independent, and network-independent protocol that runs on Cisco devices and enables networking applications to learn about directly connected devices nearby. This protocol facilitates the management of Cisco devices by discovering these devices, determining how they are configured, and allowing systems using different network-layer protocols to learn about each other.

This module describes Cisco Discovery Protocol Version 2 and how it functions with SNMP.

Information about Cisco Discovery Protocol

The following sections provide information about Cisco Discovery Protocol

Default Cisco Discovery Protocol Configuration

This table shows the default Cisco Discovery Protocol configuration.

Feature	Default Setting
Cisco Discovery Protocol global state	Enabled
Cisco Discovery Protocol interface state	Enabled
Cisco Discovery Protocol timer (packet update frequency)	60 seconds
Cisco Discovery Protocol holdtime (before discarding)	180 seconds
Cisco Discovery Protocol Version-2 advertisements	Enabled

Cisco Discovery Protocol Overview

Cisco Discovery Protocol is a device discovery protocol that runs over Layer 2 (the data-link layer) on all Cisco-manufactured devices (routers, bridges, access servers, controllers, and switches) and allows network management applications to discover Cisco devices that are neighbors of already known devices. With Cisco Discovery Protocol, network management applications can learn the device type and the SNMP agent address

of neighboring devices running lower-layer, transparent protocols. This feature enables applications to send SNMP queries to neighboring devices.

Cisco Discovery Protocol runs on all media that support Subnetwork Access Protocol (SNAP). Because Cisco Discovery Protocol runs over the data-link layer only, two systems that support different network-layer protocols can learn about each other.

Each Cisco Discovery Protocol-configured device sends periodic messages to a multicast address, advertising at least one address at which it can receive SNMP messages. The advertisements also contain time-to-live, or holdtime information, which is the length of time a receiving device holds Cisco Discovery Protocol information before discarding it. Each device also listens to the messages sent by other devices to learn about neighboring devices.

On the device, Cisco Discovery Protocol enables Network Assistant to display a graphical view of the network. The device uses Cisco Discovery Protocol to find cluster candidates and maintain information about cluster members and other devices up to three cluster-enabled devices away from the command device by default.

The following applies to a device and connected endpoint devices:

- Cisco Discovery Protocol identifies connected endpoints that communicate directly with the device.
- To prevent duplicate reports of neighboring devices, only one wired device reports the location information.
- The wired device and the endpoints both send and receive location information.

How to Configure Cisco Discovery Protocol

The following sections provide information about how to configure Cisco Discovery Protocol.

Configuring Cisco Discovery Protocol Characteristics

You can configure these Cisco Discovery Protocol characteristics:

- Frequency of Cisco Discovery Protocol updates
- Amount of time to hold the information before discarding it
- Whether or not to send Version 2 advertisements

Note Steps 3 through 5 are all optional and can be performed in any order.

Follow these steps to configure the Cisco Discovery Protocol characteristics.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	
Step 3	**cdp timer** *seconds* **Example:** `Device(config)# cdp timer 20`	(Optional) Sets the transmission frequency of Cisco Discovery Protocol updates in seconds. The range is 5 to 254; the default is 60 seconds.
Step 4	**cdp holdtime** *seconds* **Example:** `Device(config)# cdp holdtime 60`	(Optional) Specifies the amount of time a receiving device should hold the information sent by your device before discarding it. The range is 10 to 255 seconds; the default is 180 seconds.
Step 5	**cdp advertise-v2** **Example:** `Device(config)# cdp advertise-v2`	(Optional) Configures Cisco Discovery Protocol to send Version 2 advertisements. This is the default state.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

Use the **no** form of the Cisco Discovery Protocol commands to return to the default settings.

Disabling Cisco Discovery Protocol

Device clusters and other Cisco devices (such as Cisco IP Phones) regularly exchange Cisco Discovery Protocol messages. Disabling Cisco Discovery Protocol can interrupt cluster discovery and device connectivity.

Note Cisco Discovery Protocol is enabled by default.

Follow these steps to disable the Cisco Discovery Protocol device discovery capability.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no cdp run** **Example:** `Device(config)# no cdp run`	Disables Cisco Discovery Protocol.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

You must reenable Cisco Discovery Protocol to use it.

Enabling Cisco Discovery Protocol

Cisco Discovery Protocol is enabled by default.

Note Device clusters and other Cisco devices (such as Cisco IP Phones) regularly exchange Cisco Discovery Protocol messages. Disabling Cisco Discovery Protocol can interrupt cluster discovery and device connectivity.

Follow these steps to enable Cisco Discovery Protocol when it has been disabled.

Before you begin

Cisco Discovery Protocol must be disabled, or it cannot be enabled.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**cdp run** **Example:** `Device(config)# cdp run`	Enables Cisco Discovery Protocol if it has been disabled.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.

	Command or Action	Purpose
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

Use the **show run all** command to check if Cisco Discovery Protocol has been enabled. If you run the **show run** command, the enabling of Cisco Discovery Protocol may not be displayed.

Disabling Cisco Discovery Protocol on an Interface

Device clusters and other Cisco devices (such as Cisco IP Phones) regularly exchange Cisco Discovery Protocol messages. Disabling Cisco Discovery Protocol can interrupt cluster discovery and device connectivity.

Note

- Cisco Discovery Protocol is enabled by default on all supported interfaces to send and to receive Cisco Discovery Protocol information.
- Cisco Discovery Protocol bypass is not supported and may cause a port go into err-disabled state.

Follow these steps to disable Cisco Discovery Protocol on a port.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface on which you are disabling Cisco Discovery Protocol, and enters interface configuration mode.
Step 4	**no cdp enable** **Example:**	Disables Cisco Discovery Protocol on the interface specified in Step 3.

	Command or Action	Purpose
	`Device(config-if)# no cdp enable`	
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Enabling Cisco Discovery Protocol on an Interface

Device clusters and other Cisco devices (such as Cisco IP Phones) regularly exchange Cisco Discovery Protocol messages. Disabling Cisco Discovery Protocol can interrupt cluster discovery and device connectivity.

Note

- Cisco Discovery Protocol is enabled by default on all supported interfaces to send and to receive Cisco Discovery Protocol information.
- Cisco Discovery Protocol bypass is not supported and may cause a port go into err-disabled state.

Follow these steps to enable Cisco Discovery Protocol on a port on which it has been disabled.

Before you begin

Cisco Discovery Protocol must be disabled on the port that you are trying to Cisco Discovery Protocol enable on, or it cannot be enabled.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/1`	Specifies the interface on which you are enabling Cisco Discovery Protocol, and enters interface configuration mode.
Step 4	**cdp enable** **Example:** `Device(config-if)# cdp enable`	Enables Cisco Discovery Protocol on a disabled interface.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring and Maintaining Cisco Discovery Protocol

Table 160: Commands for Displaying Cisco Discovery Protocol Information

Command	Description
clear cdp counters	Resets the traffic counters to zero.
clear cdp table	Deletes the Cisco Discovery Protocol table of information about neighbors.
show cdp	Displays global information, such as frequency of transmissions and the holdtime for packets being sent.

Command	Description
show cdp entry *entry-name* [**version**] [**protocol**]	Displays information about a specific neighbor. You can enter an asterisk (*) to display all Cisco Discovery Protocol neighbors, or you can enter the name of the neighbor about which you want information. You can also limit the display to information about the protocols enabled on the specified neighbor or information about the version of software running on the device.
show cdp interface [*interface-id*]	Displays information about interfaces where Cisco Discovery Protocol is enabled. You can limit the display to the interface about which you want information.
show cdp neighbors [*interface-id*] [*detail*]	Displays information about neighbors, including device type, interface type and number, holdtime settings, capabilities, platform, and port ID. You can limit the display to neighbors of a specific interface or expand the display to provide more detailed information.
show cdp traffic	Displays Cisco Discovery Protocol counters, including the number of packets sent and received and checksum errors.
show ap cdp neighbors	Displays information regarding the access point's Cisco Discovery Protocol neighbors.
show ap cdp neighbors detail	Displays detailed information regarding the access point's Cisco Discovery Protocol neighbors.
show ap name *ap-name* **cdp neighbors**	Displays the Cisco Discovery Protocol information for an access point.
show ap name *ap-name* **cdp neighbors detail**	Displays details about a specific access point neighbor that is using Cisco Discovery Protocol.

CHAPTER 160

Configuring Cisco Discovery Protocol Bypass

In Cisco Discovery Protocol Bypass mode Cisco Discovery Protocol packets are received and transmitted unchanged. Received packets are not processed. No packets are generated. In this mode, 'bump-in-the-wire' behavior is applied to Cisco Discovery Protocol packets. This is a backward compatible mode, equivalent to not having Cisco Discovery Protocol support.

Restrictions for Cisco Discovery Protocol Bypass

Cisco Discovery Protocol Bypass does not support standard ACLs on the switch port.

Information about Cisco Discovery Protocol Bypass

When a Cisco IP Phone is plugged into a port that is configured with a Voice VLAN and single-host mode, the phone will be silently allowed onto the network by way of a feature known as Cisco Discovery Protocol Bypass. The phone (or any device) that sends the appropriate Type Length Value (TLV) in a Cisco Discovery Protocol message will be allowed access to the voice VLAN.

In Cisco Discovery Protocol Bypass mode, Cisco Disocvery Protocol packets are received and transmitted unchanged. Received packets are not processed. No packets are generated. In this mode, 'bump-in-the-wire' behaviour is applied to Cisco Discovery Protocol packets. This is a backward compatible mode, equivalent to not having Cisco Discovery Protocol support.

In Cisco Discovery Protocol Bypass mode authentication sessions are established in single and multi-host modes for IP Phones. However, if voice VLAN and 802.1x on an interface port is enabled, then Cisco Discovery Protocol Bypass is enabled when the host mode is set to single or multi-host mode.

It is possible to use the Multi-Domain Authentication (MDA) feature instead of Cisco Discovery Protocol Bypass feature as it provides better Access Control, Visibility and Authorization.

Note By default the host mode is set to single mode in legacy mode and multi-authentication in the edge mode.

Cisco Discovery Protocol Enhancement for Second Port Disconnect—Allows a Cisco IP phone to send a Cisco Discovery Protocol message to the switch when a host unplugs from behind the phone. The switch is then able to clear any authenticated session for the indirectly connected host, the same as if the host had been directly connected and the switch had detected a link down event. This is supported in latest IP telephones.

Cisco Discovery Protocol Bypass provides no support for third-party phones—Cisco Discovery Protocol Bypass works only with Cisco phones.

How to configure Cisco Discovery Protocol Bypass

Follow these steps to enable Cisco Discovery Protocol Bypass:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `# configure terminal`	Enters the global configuration mode.
Step 3	**interface** *interface-id* **Example:** `(config)# interface gigabitethernet 1/1`	Specifies a physical port, and enters interface configuration mode. • Valid interfaces are physical ports.
Step 4	**switchport mode access** **Example:** `(config-if)# switchport mode access`	Specifies that the interface is in access mode.
Step 5	**switchport access vlan** *vlan id* **Example:** `(config-if)# switchport access vlan 10`	Assigns all ports as static-access ports in the same VLAN • If you configure the port as a static-access port, assign it to only one VLAN. The range is 1 to 4094.
Step 6	**switchport voice vlan** *vlan-id* **Example:** `(config-if)# switchport voice vlan 3`	Instruct the Cisco IP phone to forward all voice traffic through the specified VLAN. By default, the Cisco IP phone forwards the voice traffic with an 802.1Q priority of 5. Valid VLAN IDs are from 1 to 4094 when the enhanced software image (EI) is installed and 1 to 1001 when the standard software image is installed. Do not enter leading zeros.

	Command or Action	Purpose
Step 7	**authentication port-control auto** **Example:** `(config-if)# authentication port-control auto`	Enables 802.1x authentication on the port.
Step 8	**authentication host-mode** {**single-host** | **multi-host**} **Example:** `(config-if)# authentication host-mode single | multi-host`	The keywords allow the following: • single-host-Single host (client) on an IEEE 802.1X-authorized port. • multi-host-Multiple hosts on an 802.1X-authorized port after a authenticating a single host.
Step 9	**dot1x pae authenticator** **Example:** `(config-if)# dot1x pae authenticator`	Enables 802.1X authentication on the port with default parameters

Configuration Examples for Cisco Discovery Protocol Bypass

Example: Enabling Cisco Discovery Protocol Bypass

Cisco Discovery Protocol Bypass is enabled by default once 'Authetication port-control auto' is configured with dotx1 or MAB or if voice vlan is configured on interface along with single/multiple host mode.

This following configuration example configures Cisco Disovery Protocol Bypass when authenticating using MAB.

```
 (config)# interface gigabitethernet 1/1
(config-if)# switchport mode access
(config-if)# switchport access vlan 10
(config-if)# switchport voice vlan 3
(config-if)# authentication port-control auto
(config-if)# mab
```

Displaying Cisco Discovery Protocol neighbours

The following configuration example displays Cisco Discovery Protocol neighbours.

```
# show cdp neighbors gigabitethernet 1/1 detail
Device ID: SEP24B657B038DF
Entry address(es):
Platform: Cisco IP Phone 9971,  Capabilities: Host Phone Two-port Mac Relay
Interface: gigabitethernet 1/1,  Port ID (outgoing port): Port 1
Holdtime : 157 sec
Second Port Status: Down <<<<<<<<<<
Version :
```

```
sip9971.9-1-1SR1
advertisement version: 2
Duplex: full
Power drawn: 12.804 Watts
Power request id: 57146, Power management id: 4
Power request levels are:12804 0 0 0 0
Total cdp entries displayed : 1
```

Example:Disabling Cisco Discovery Protocol Bypass

To disable Cisco Discovery Protocol Bypass,‘Authetication port-control auto’ needs to be removed from the interface.

CHAPTER 161

Configuring Simple Network Management Protocol

Prerequisites for SNMP

Supported SNMP Versions

This software release supports the following SNMP versions:

- SNMPv1—The Simple Network Management Protocol, a Full Internet Standard, defined in RFC 1157.
- SNMPv2C replaces the Party-based Administrative and Security Framework of SNMPv2Classic with the community-string-based Administrative Framework of SNMPv2C while retaining the bulk retrieval and improved error handling of SNMPv2Classic. It has these features:
  - SNMPv2—Version 2 of the Simple Network Management Protocol, a Draft Internet Standard, defined in RFCs 1902 through 1907.
  - SNMPv2C—The community-string-based Administrative Framework for SNMPv2, an Experimental Internet Protocol defined in RFC 1901.
- SNMPv3—Version 3 of the SNMP is an interoperable standards-based protocol defined in RFCs 2273 to 2275. SNMPv3 provides secure access to devices by authenticating and encrypting packets over the network and includes these security features:
  - Message integrity—Ensures that a packet was not tampered with in transit.
  - Authentication—Determines that the message is from a valid source.
  - Encryption—Mixes the contents of a package to prevent it from being read by an unauthorized source.

Note To select encryption, enter the **priv** keyword.

Both SNMPv1 and SNMPv2C use a community-based form of security. The community of managers able to access the agent's MIB is defined by an IP address access control list and password.

SNMPv2C includes a bulk retrieval function and more detailed error message reporting to management stations. The bulk retrieval function retrieves tables and large quantities of information, minimizing the number of round-trips required. The SNMPv2C improved error-handling includes expanded error codes that distinguish different kinds of error conditions; these conditions are reported through a single error code in SNMPv1. Error return codes in SNMPv2C report the error type.

SNMPv3 provides for both security models and security levels. A security model is an authentication strategy set up for a user and the group within which the user resides. A security level is the permitted level of security within a security model. A combination of the security level and the security model determine which security method is used when handling an SNMP packet. Available security models are SNMPv1, SNMPv2C, and SNMPv3.

The following table identifies characteristics and compares different combinations of security models and levels:

Table 161: SNMP Security Models and Levels

Model	Level	Authentication	Encryption	Result
SNMPv1	noAuthNoPriv	Community string	No	Uses a community string match for authentication.
SNMPv2C	noAuthNoPriv	Community string	No	Uses a community string match for authentication.
SNMPv3	noAuthNoPriv	Username	No	Uses a username match for authentication.
SNMPv3	authNoPriv	Message Digest 5 (MD5) or Secure Hash Algorithm (SHA)	No	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms.

Model	Level	Authentication	Encryption	Result
SNMPv3	authPriv	MD5 or SHA	Data Encryption Standard (DES) or Advanced Encryption Standard (AES)	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms. Allows specifying the User-based Security Model (USM) with these encryption algorithms: • DES 56-bit encryption in addition to authentication based on the CBC-DES (DES-56) standard. • 3DES 168-bit encryption • AES 128-bit, 192-bit, or 256-bit encryption

You must configure the SNMP agent to use the SNMP version supported by the management station. Because an agent can communicate with multiple managers, you can configure the software to support communications using SNMPv1, SNMPv2C, or SNMPv3.

Restrictions for SNMP

Version Restrictions

- SNMPv1 does not support informs.
- To avoid SNMPv3 authentication failure, you should manually configure SNMP engineID on the device before SNMPv3 user configuration. With this, the user can manage and administer the device as the user is tied to the engineID.

Information About SNMP

The following sections provide information about SNMP.

SNMP Overview

SNMP is an application-layer protocol that provides a message format for communication between managers and agents. The SNMP system consists of an SNMP manager, an SNMP agent, and a management information base (MIB). The SNMP manager can be part of a network management system (NMS). The agent and MIB reside on the device. To configure SNMP on the device, you define the relationship between the manager and the agent.

The SNMP agent contains MIB variables whose values the SNMP manager can request or change. A manager can get a value from an agent or store a value into the agent. The agent gathers data from the MIB, the repository for information about device parameters and network data. The agent can also respond to a manager's requests to get or set data.

An agent can send unsolicited traps to the manager. Traps are messages alerting the SNMP manager to a condition on the network. Traps can mean improper user authentication, restarts, link status (up or down), MAC address tracking, closing of a TCP connection, loss of connection to a neighbor, or other significant events.

SNMP Manager Functions

The SNMP manager uses information in the MIB to perform the operations described in the following table:

Table 162: SNMP Operations

Operation	Description
get-request	Retrieves a value from a specific variable.
get-next-request	Retrieves a value from a variable within a table.[20]
get-bulk-request[21]	Retrieves large blocks of data, such as multiple rows in a table, that would otherwise require the transmission of many small blocks of data.
get-response	Replies to a get-request, get-next-request, and set-request sent by an NMS.
set-request	Stores a value in a specific variable.
trap	An unsolicited message sent by an SNMP agent to an SNMP manager when some event has occurred.

[20] With this operation, an SNMP manager does not need to know the exact variable name. A sequential search is performed to find the needed variable from within a table.

[21] The get-bulk command only works with SNMPv2 or later.

Note We recommend that the SNMP Manager exclude the **ciscoFlashFileDate** MIB object from its query, to avoid performance related issues. This is because, though the **ciscoFlashFileDate** object is published in the MIB, it is not supported on the product.

SNMP Agent Functions

The SNMP agent can receive requests from one or more SNMP managers. Every request carries the NMS IP address, the number of times an NMS polls the agent, and a timestamp of polling. This information can be tracked for both IPv4 and IPv6 servers.

The SNMP agent responds to SNMP manager requests as follows:

- Get a MIB variable—The SNMP agent begins this function in response to a request from the NMS. The agent retrieves the value of the requested MIB variable and responds to the NMS with that value.
- Set a MIB variable—The SNMP agent begins this function in response to a message from the NMS. The SNMP agent changes the value of the MIB variable to the value requested by the NMS.

Use the **show snmp stats hosts** command to display the list of the SNMP managers requests in the queue, and use the **clear snmp stats hosts** command to clear the queue.

The SNMP agent also sends unsolicited trap messages to notify an NMS that a significant event has occurred on the agent. Examples of trap conditions include, but are not limited to, when a port or module goes up or down, when spanning-tree topology changes occur, and when authentication failures occur.

SNMP Community Strings

SNMP community strings authenticate access to MIB objects and function as embedded passwords. In order for the NMS to access the device, the community string definitions on the NMS must match at least one of the three community string definitions on the device.

A community string can have one of the following attributes:

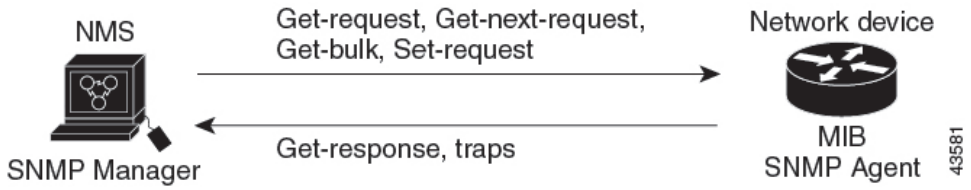
- Read-only (RO)—Gives all objects in the MIB except the community strings read access to authorized management stations, but does not allow write access.
- Read-write (RW)—Gives all objects in the MIB read and write access to authorized management stations, but does not allow access to the community strings.
- When a cluster is created, the command device manages the exchange of messages among member devices and the SNMP application. The Network Assistant software appends the member device number (@esN, where N is the device number) to the first configured RW and RO community strings on the command device and propagates them to the member devices.

SNMP MIB Variables Access

An NMS software uses the device MIB variables to set device variables and to poll devices on the network for specific information. The results of a poll can be displayed as a graph and analyzed to troubleshoot internetworking problems, increase network performance, verify the configuration of devices, monitor traffic loads, and more.

As shown in the figure, the SNMP agent gathers data from the MIB. The agent can send traps, or notification of certain events, to the SNMP manager, which receives and processes the traps. Traps alert the SNMP manager to a condition on the network such as improper user authentication, restarts, link status (up or down), MAC address tracking, and so forth. The SNMP agent also responds to MIB-related queries sent by the SNMP manager in *get-request*, *get-next-request*, and *set-request* format.

Figure 145: SNMP Network



SNMP Flash MIB

The Cisco Flash MIB is used to query flash file data from Cisco devices. Earlier the number of files listed per partition per device in the Flash MIB was limited to 100. The limitation of Flash MIB listing 100 files per partition per device has been removed. Now Flash MIB will fetch all the files from the flash file system. It is mandatory to use the **snmp mib flash cache** command to perform a Flash MIB walk. The **snmp mib flash cache** command will prefetch all the files into the local Flash MIB cache. Since the 100 file limitation has been removed the Flash MIB walk to retrieve the files will take longer.

Note It is recommended that you do not to use the **snmp mib flash cache** command on the switch due to its impact on the CPU. To prevent SNMP walks from timing out we recommend the snmp walk to have a minimum timeout period of 10 seconds and a default retry of 5 seconds.

SNMP Notifications

SNMP allows the device to send notifications to SNMP managers when particular events occur. SNMP notifications can be sent as traps or inform requests. In command syntax, unless there is an option in the command to select either traps or informs, the keyword traps refers to either traps or informs, or both. Use the **snmp-server host** command to specify whether to send SNMP notifications as traps or informs.

Note SNMPv1 does not support informs.

Traps are unreliable because the receiver does not send an acknowledgment when it receives a trap, and the sender cannot determine if the trap was received. When an SNMP manager receives an inform request, it acknowledges the message with an SNMP response protocol data unit (PDU). If the sender does not receive a response, the inform request can be sent again. Because they can be resent, informs are more likely than traps to reach their intended destination.

The characteristics that make informs more reliable than traps also consume more resources in the device and in the network. Unlike a trap, which is discarded as soon as it is sent, an inform request is held in memory until a response is received or the request times out. Traps are sent only once, but an inform might be resent or retried several times. The retries increase traffic and contribute to a higher overhead on the network. Therefore, traps and informs require a trade-off between reliability and resources. If it is important that the SNMP manager receive every notification, use inform requests. If traffic on the network or memory in the device is a concern and notification is not required, use traps.

SNMP ifIndex MIB Object Values

The SNMP agent's IF-MIB module comes up shortly after reboot. As various physical interface drivers are initialized they register with the IF-MIB module, essentially saying "Give me an ifIndex number". The IF-MIB module assigns the next available ifIndex number on a first-come-first-served basis. That is, minor differences in driver initialization order from one reboot to another can result in the same physical interface getting a different ifIndex number than it had before the reboot (unless ifIndex persistency is enabled of course).

SNMP ENTITY-MIB Identifiers

ENTITY-MIB contains information for managing physical entities such as field-replaceable units (FRUs) on a device. Each entity is identified by a unique index number-*entPhysicalIndex* that is used to access information about the entity in current and other MIBs. An online insertion and removal (OIR) of the entity results in the entity being assigned the next available *entPhysicalIndex* number, irrespective of whether a new entity is inserted or an existing entity is reinserted.

SNMP and Syslog Over IPv6

To support both IPv4 and IPv6, IPv6 network management requires both IPv6 and IPv4 transports. Syslog over IPv6 supports address data types for these transports.

Simple Network Management Protocol (SNMP) and syslog over IPv6 provide these features:

- Support for both IPv4 and IPv6
- IPv6 transport for SNMP and to modify the SNMP agent to support traps for an IPv6 host
- SNMP- and syslog-related MIBs to support IPv6 addressing
- Configuration of IPv6 hosts as trap receivers

For support over IPv6, SNMP modifies the existing IP transport mapping to simultaneously support IPv4 and IPv6. These SNMP actions support IPv6 transport management:

- Opens User Datagram Protocol (UDP) SNMP socket with default settings
- Provides a new transport mechanism called *SR_IPV6_TRANSPORT*
- Sends SNMP notifications over IPv6 transport
- Supports SNMP-named access lists for IPv6 transport
- Supports SNMP proxy forwarding using IPv6 transport
- Verifies SNMP Manager feature works with IPv6 transport

For information on SNMP over IPv6, including configuration procedures, see the “Managing Cisco IOS Applications over IPv6” chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

For information about syslog over IPv6, including configuration procedures, see the “Implementing IPv6 Addressing and Basic Connectivity” chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

Default SNMP Configuration

Feature	Default Setting
SNMP agent	Disabled[22].
SNMP trap receiver	None configured.
SNMP traps	None enabled except the trap for TCP connections (tty).
SNMP version	If no version keyword is present, the default is Version 1.
SNMPv3 authentication	If no keyword is entered, the default is the **noauth** (noAuthNoPriv) security level.
SNMP notification type	If no type is specified, all notifications are sent.

[22] This is the default when the device starts and the startup configuration does not have any **snmp-server** global configuration commands.

SNMP Configuration Guidelines

The device requires one of the following global configuration commands configured in order to open SNMP UDP ports 161 and 162 and enable the SNMP agent: **snmp-server host**, or **snmp-server user**, or **snmp-server community**, or **snmp-server manager**.

An SNMP *group* is a table that maps SNMP users to SNMP views. An SNMP *user* is a member of an SNMP group. An SNMP *host* is the recipient of an SNMP trap operation. An SNMP *engine ID* is a name for the local or remote SNMP engine.

When configuring SNMP, follow these guidelines:

- When configuring an SNMP group, do not specify a notify view. The **snmp-server host** global configuration command auto-generates a notify view for the user and then adds it to the group associated with that user. Modifying the group's notify view affects all users associated with that group.
- To configure a remote user, specify the IP address or port number for the remote SNMP agent of the device where the user resides.
- Before you configure remote users for a particular agent, configure the SNMP engine ID, using the **snmp-server engineID** global configuration command with the **remote** option. The remote agent's SNMP engine ID and user password are used to compute the authentication and privacy digests. If you do not configure the remote engine ID first, the configuration command fails.
- When configuring SNMP informs, you need to configure the SNMP engine ID for the remote agent in the SNMP database before you can send proxy requests or informs to it.
- If a local user is not associated with a remote host, the device does not send informs for the **auth** (authNoPriv) and the **priv** (authPriv) authentication levels.
- Changing the value of the SNMP engine ID has significant results. A user's password (entered on the command line) is converted to an MD5 or SHA security digest based on the password and the local engine ID. The command-line password is then destroyed, as required by RFC 2274. Because of this deletion, if the value of the engine ID changes, the security digests of SNMPv3 users become invalid, and you need to reconfigure SNMP users by using the **snmp-server user** *username* global configuration command. Similar restrictions require the reconfiguration of community strings when the engine ID changes.

- When you configure the SNMP server host with the default UDP port, 162, the output of the **show running-config** command does not display the UDP port value. If you specify a UDP port value other than the default by using the **snmp-server host** {*host-addr*} *community-string* **udp-port** *value* command, the UDP port number will be displayed in the **show running-config** command output. You can configure the **snmp-server host** command with or without the default UDP port 162; however, you cannot configure both simultaneously.

  The following examples are correct:

```
Device(config)# snmp-server host 10.10.10.10 community udp-port 163
Device(config)# snmp-server host 10.10.10.10 community

Device(config)# snmp-server host 10.10.10.10 community udp-port 163
Device(config)# snmp-server host 10.10.10.10 community udp-port 162
```

  The following examples are incorrect:

```
Device(config)# snmp-server host 10.10.10.10 community udp-port 163
Device(config)# snmp-server host 10.10.10.10 community
Device(config)# snmp-server host 10.10.10.10 community udp-port 162

Device(config)# snmp-server host 10.10.10.10 community udp-port 163
Device(config)# snmp-server host 10.10.10.10 community udp-port 162
Device(config)# snmp-server host 10.10.10.10 community
```

How to Configure SNMP

The following sections provide information on how to configure SNMP.

SNMP Community Strings

SNMP community strings authenticate access to MIB objects and function as embedded passwords. In order for the NMS to access the device, the community string definitions on the NMS must match at least one of the three community string definitions on the device.

A community string can have one of the following attributes:

- Read-only (RO)—Gives all objects in the MIB except the community strings read access to authorized management stations, but does not allow write access.
- Read-write (RW)—Gives all objects in the MIB read and write access to authorized management stations, but does not allow access to the community strings.
- When a cluster is created, the command device manages the exchange of messages among member devices and the SNMP application. The Network Assistant software appends the member device number (@esN, where N is the device number) to the first configured RW and RO community strings on the command device and propagates them to the member devices.

Configuring SNMP Groups and Users

You can specify an identification name (engine ID) for the local or remote SNMP server engine on the device. You can configure an SNMP server group that maps SNMP users to SNMP views, and you can add new users to the SNMP group.

Follow these steps to configure SNMP groups and users on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server engineID** {**local** *engineid-string* | **remote** *ip-address* [**udp-port** *port-number*] *engineid-string*} **Example:** `Device(config)# snmp-server engineID local 1234`	Configures a name for either the local or remote copy of SNMP. • The *engineid-string* is a 24-character ID string with the name of the copy of SNMP. You need not specify the entire 24-character engine ID if it has trailing zeros. Specify only the portion of the engine ID up to the point where only zeros remain in the value. The Step Example configures an engine ID of 123400000000000000000000. • If you select **remote**, specify the *ip-address* of the device that contains the remote copy of SNMP and the optional User Datagram Protocol (UDP) port on the remote device. The default is 162.
Step 4	**snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**read** *readview*] [**write** *writeview*] [**notify** *notifyview*] [**access** *access-list*] **Example:** `Device(config)# snmp-server group public v2c access lmnop`	Configures a new SNMP group on the remote device. For *group-name*, specify the name of the group. Specify one of the following security models: • **v1** is the least secure of the possible security models. • **v2c** is the second least secure model. It allows transmission of informs and integers twice the normal width. • **v3**, the most secure, requires you to select one of the following authentication levels:

	Command or Action	Purpose
		auth—Enables the Message Digest 5 (MD5) and the Secure Hash Algorithm (SHA) packet authentication. **noauth**—Enables the noAuthNoPriv security level. This is the default if no keyword is specified. **priv**—Enables Data Encryption Standard (DES) packet encryption (also called privacy). (Optional) Enter **read** *readview* with a string (not to exceed 64 characters) that is the name of the view in which you can only view the contents of the agent. (Optional) Enter **write** *writeview* with a string (not to exceed 64 characters) that is the name of the view in which you enter data and configure the contents of the agent. (Optional) Enter **notify** *notifyview* with a string (not to exceed 64 characters) that is the name of the view in which you specify a notify, inform, or trap. (Optional) Enter **access** *access-list* with a string (not to exceed 64 characters) that is the name of the access list.
Step 5	**snmp-server user** *username group-name* {**remote** *host* [**udp-port** *port*] } {**v1** [**access** *access-list*] | **v2c** [**access** *access-list*] | **v3** [**encrypted**] [**access** *access-list*] [**auth** {**md5** | **sha**} *auth-password*] } [*priv* {**des** | **3des** | **aes** {**128** | **192** | **256**}} *priv-password*] **Example:** `Device(config)# snmp-server user Pat public v2c`	Adds a new user for an SNMP group. The *username* is the name of the user on the host that connects to the agent. The *group-name* is the name of the group to which the user is associated. Enter **remote** to specify a remote SNMP entity to which the user belongs and the hostname or IP address of that entity with the optional UDP port number. The default is 162. Enter the SNMP version number (**v1**, **v2c**, or **v3**). If you enter **v3**, you have these additional options: • **encrypted** specifies that the password appears in encrypted format. This keyword is available only when the **v3** keyword is specified. • **auth** is an authentication level setting session that can be either the

	Command or Action	Purpose
		HMAC-MD5-96 (**md5**) or the HMAC-SHA-96 (**sha**) authentication level and requires a password string *auth-password* (not to exceed 64 characters). If you enter **v3** you can also configure a private (**priv**) encryption algorithm and password string *priv-password* using the following keywords (not to exceed 64 characters): • **priv** specifies the User-based Security Model (USM). • **des** specifies the use of the 56-bit DES algorithm. • **3des** specifies the use of the 168-bit DES algorithm. • **aes** specifies the use of the DES algorithm. You must select either 128-bit, 192-bit, or 256-bit encryption. (Optional) Enter **access** *access-list* with a string (not to exceed 64 characters) that is the name of the access list. **Note** The algorithms — **md5**, **des**, **3des** is not supported in a SNMPv3 group when the compliance shield is disabled. You need to enable the compliance shield using the **crypto engine compliance shield enable** command and reboot the device to configure the algorithms — **md5**, **des** and **3des**.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:**	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`Device# copy running-config startup-config`	

Opening or Closing SNMP UDP Ports

The SNMP process uses ports 161 and 162 where port 161 is used for polling the device and port 162 is used for sending notifications from the agent to the server. The SNMP UDP ports remain closed unless one of the requisite commands is configured. This design provides additional security by opening the ports only when needed and prevents a device from listening to a port unnecessarily.

Beginning in user EXEC mode, follow these steps to open the SNMP UDP ports.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server** {**host** | **user** | **community** | **manager**} **Example:** `Device(config)# snmp-server host`	Opens SNMP UDP ports 161 and 162. Configuring any one of the options (**host**, **user**, **community**, **manager**) opens both ports. To close the ports, enter the **no** form of all the options that you have configured. The ports remain open as long as even one of the keywords is configured. If you enter the **no snmp-server** command, without any of the keywords, the SNMP process is shut down and not just the SNMP UDP ports.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show udp** **Example:** `Device# show udp`	Displays the SNMP UDP ports. If one of the requisite commands is configured, ports 161 and 162 will display value `listen` under the remote field.
Step 6	**copy running-config startup-config** **Example:**	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`Device# copy running-config startup-config`	

SNMP Notifications

SNMP allows the device to send notifications to SNMP managers when particular events occur. SNMP notifications can be sent as traps or inform requests. In command syntax, unless there is an option in the command to select either traps or informs, the keyword traps refers to either traps or informs, or both. Use the **snmp-server host** command to specify whether to send SNMP notifications as traps or informs.

Note SNMPv1 does not support informs.

Traps are unreliable because the receiver does not send an acknowledgment when it receives a trap, and the sender cannot determine if the trap was received. When an SNMP manager receives an inform request, it acknowledges the message with an SNMP response protocol data unit (PDU). If the sender does not receive a response, the inform request can be sent again. Because they can be resent, informs are more likely than traps to reach their intended destination.

The characteristics that make informs more reliable than traps also consume more resources in the device and in the network. Unlike a trap, which is discarded as soon as it is sent, an inform request is held in memory until a response is received or the request times out. Traps are sent only once, but an inform might be resent or retried several times. The retries increase traffic and contribute to a higher overhead on the network. Therefore, traps and informs require a trade-off between reliability and resources. If it is important that the SNMP manager receive every notification, use inform requests. If traffic on the network or memory in the device is a concern and notification is not required, use traps.

Setting the Agent Contact and Location Information

Follow these steps to set the system contact and location of the SNMP agent so that these descriptions can be accessed through the configuration file.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**snmp-server contact** *text* **Example:** `Device(config)# snmp-server contact Dial System Operator at beeper 21555`	Sets the system contact string.
Step 4	**snmp-server location** *text* **Example:** `Device(config)# snmp-server location Building 3/Room 222`	Sets the system location string.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Limiting TFTP Servers Used Through SNMP

Follow these steps to limit the TFTP servers used for saving and loading configuration files through SNMP to the servers specified in an access list.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**snmp-server tftp-server-list** *access-list-number* **Example:** `Device(config)# snmp-server tftp-server-list 44`	Limits the TFTP servers used for configuration file copies through SNMP to the servers in the access list. For *access-list-number*, enter an IP standard access list numbered from 1 to 99 and 1300 to 1999.
Step 4	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] **Example:** `Device(config)# access-list 44 permit 10.1.1.2`	Creates a standard access list, repeating the command as many times as necessary. For *access-list-number*, enter the access list number specified in Step 3. The **deny** keyword denies access if the conditions are matched. The **permit** keyword permits access if the conditions are matched. For *source*, enter the IP address of the TFTP servers that can access the device. (Optional) For *source-wildcard*, enter the wildcard bits, in dotted decimal notation, to be applied to the source. Place ones in the bit positions that you want to ignore. The access list is always terminated by an implicit deny statement for everything.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Disabling the SNMP Agent

The **no snmp-server** global configuration command disables all running versions (Version 1, Version 2C, and Version 3) of the SNMP agent on the device and shuts down the SNMP process. You reenable all versions

of the SNMP agent by entering one of the following commands in global configuration mode: **snmp-server host**, or **snmp-server user**, or **snmp-server community**, or **snmp-server manager**. There is no Cisco IOS command specifically designated for enabling SNMP.

Follow these steps to disable the SNMP agent.

Before you begin

The SNMP Agent must be enabled before it can be disabled. The SNMP agent is enabled by the first **snmp-server** global configuration command entered on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no snmp-server** **Example:** `Device(config)# no snmp-server`	Disables the SNMP agent operation.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

SNMP Examples

This example shows how to enable all versions of SNMP. The configuration permits any SNMP manager to access all objects with read-only permissions using the community string *public*. This configuration does not cause the device to send any traps.

```
Device(config)# snmp-server community public
```

This example shows how to permit any SNMP manager to access all objects with read-only permission using the community string *public*. The device also sends VTP traps to the hosts 192.180.1.111 and 192.180.1.33 using SNMPv1 and to the host 192.180.1.27 using SNMPv2C. The community string *public* is sent with the traps.

```
Device(config)# snmp-server community public
Device(config)# snmp-server enable traps vtp
Device(config)# snmp-server host 192.180.1.27 version 2c public
Device(config)# snmp-server host 192.180.1.111 version 1 public
Device(config)# snmp-server host 192.180.1.33 public
```

This example shows how to allow read-only access for all objects to members of access list 4 that use the *comaccess* community string. No other SNMP managers have access to any objects. SNMP Authentication Failure traps are sent by SNMPv2C to the host *cisco.com* using the community string *public*.

```
Device(config)# snmp-server community comaccess ro 4
Device(config)# snmp-server enable traps snmp authentication
Device(config)# snmp-server host cisco.com version 2c public
```

This example shows how to send Entity MIB traps to the host *cisco.com*. The community string is restricted. The first line enables the device to send Entity MIB traps in addition to any traps previously enabled. The second line specifies the destination of these traps and overwrites any previous **snmp-server** host commands for the host *cisco.com*.

```
Device(config)# snmp-server enable traps entity
Device(config)# snmp-server host cisco.com restricted entity
```

This example shows how to enable the device to send all traps to the host *myhost.cisco.com* using the community string *public*:

```
Device(config)# snmp-server enable traps
Device(config)# snmp-server host myhost.cisco.com public
```

This example shows how to associate a user with a remote host and to send **auth** (authNoPriv) authentication-level informs when the user enters global configuration mode:

```
Device(config)# snmp-server engineID remote 192.180.1.27 00000063000100a1c0b4011b
Device(config)# snmp-server group authgroup v3 auth
Device(config)# snmp-server user authuser authgroup remote 192.180.1.27 v3 auth md5 mypassword
Device(config)# snmp-server user authuser authgroup v3 auth md5 mypassword
Device(config)# snmp-server host 192.180.1.27 informs version 3 auth authuser config
Device(config)# snmp-server enable traps
Device(config)# snmp-server inform retries 0
```

This example shows how to display the entries of SNMP Managers polled to an SNMP Agent:

```
Device# show snmp stats host
Request Count                  Last Timestamp              Address
2                               00:00:01 ago               3.3.3.3
1                               1w2d ago                   2.2.2.2
```

This example shows the message displayed by the device when you configure any of the three algorithms — **md5**, **des**, **3des** in a SNMPv3 group when complaince shield is disabled:

```
Device(config)# snmp-server user md5user grp v3 auth md5 cisco1234 priv des
Sep  1 00:14:51.582 IST: %SNMP-6-AUTHPROTOCOLMD5: Authentication protocol md5 support will
 be deprecated in future
Sep  1 00:14:51.582 IST: %SNMP-6-PRIVPROTOCOLDES: Privacy protocol des support will be
deprecated in future
Sep  1 00:14:51.645 IST: %SNMP-5-WARMSTART: SNMP agent on host Switch is undergoing a warm
 start
```

This example shows the message displayed by the device when you configure any of the three algorithms — **md5**, **des**, **3des** in a SNMPv3 group when complaince shield is enabled. The crypto algorithms is supported along with a warning message:

```
Device(config)# snmp-server user md5user grp v3 auth md5 cisco1234
weaker algorithm MD5, DES and 3DES is not allowed for snmp user
```

Monitoring SNMP Status

To display SNMP input and output statistics, including the number of illegal community string entries, errors, and requested variables, use the **show snmp** privileged EXEC command. You also can use the other privileged EXEC commands listed in the table to display SNMP information.

Table 163: Commands for Displaying SNMP Information

Command	Purpose
show snmp	Displays SNMP statistics.
	Displays information on the local SNMP engine and all remote engines that have been configured on the device.
show snmp group	Displays information on each SNMP group on the network.
show snmp pending	Displays information on pending SNMP requests.
show snmp sessions	Displays information on the current SNMP sessions.
show snmp user	Displays information on each SNMP user name in the SNMP users table. **Note** You must use this command to display SNMPv3 configuration information for **auth** | **noauth** | **priv** mode. This information is not displayed in the **show running-config** output.

CHAPTER 162

Configuring Cisco IOS IP Service Level Agreements

This chapter describes how to use Cisco IOS IP Service Level Agreements (SLAs) on the switch.

Unless otherwise noted, the term *switch* refers to a standalone switch.

Restrictions on Service Level Agreements

The following are restrictions on IP SLAs network performance measurement:

- The device does not support VoIP service levels using the gatekeeper registration delay operations measurements.
- Only a Cisco IOS device can be a source for a destination IP SLAs responder.
- You cannot configure the IP SLAs responder on non-Cisco devices and Cisco IOS IP SLAs can send operational packets only to services native to those devices.

Information About Service Level Agreements

The following sections provide information about Service Level Agreements.

Cisco IOS IP Service Level Agreements (SLAs)

Cisco IOS IP SLAs send data across the network to measure performance between multiple network locations or across multiple network paths. They simulate network data and IP services and collect network performance information in real time. Cisco IOS IP SLAs generate and analyze traffic either between Cisco IOS devices or from a Cisco IOS device to a remote IP device such as a network application server. Measurements provided

by the various Cisco IOS IP SLA operations can be used for troubleshooting, for problem analysis, and for designing network topologies.

Depending on the specific Cisco IOS IP SLA operations, various network performance statistics are monitored within the Cisco device and stored in both command-line interface (CLI) and Simple Network Management Protocol (SNMP) MIBs. IP SLA packets have configurable IP and application layer options such as source and destination IP address, User Datagram Protocol (UDP)/TCP port numbers, a type of service (ToS) byte (including Differentiated Services Code Point [DSCP] and IP Prefix bits), Virtual Private Network (VPN) routing/forwarding instance (VRF), and URL web address.

Because Cisco IP SLAs are Layer 2 transport independent, you can configure end-to-end operations over disparate networks to best reflect the metrics that an end user is likely to experience. IP SLAs collect and analyze the following performance metrics:

- Delay (both round-trip and one-way)
- Jitter (directional)
- Packet loss (directional)
- Packet sequencing (packet ordering)
- Path (per hop)
- Connectivity (directional)
- Server or website download time

Because Cisco IOS IP SLA is SNMP-accessible, it can also be used by performance-monitoring applications like Cisco Prime Internetwork Performance Monitor (IPM) and other third-party Cisco partner performance management products.

Using IP SLAs can provide the following benefits:

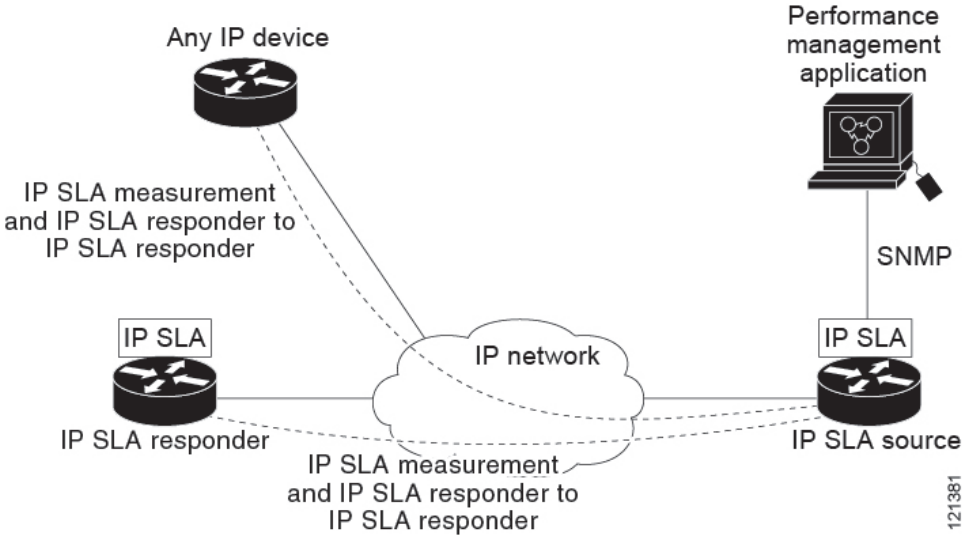
- Service-level agreement monitoring, measurement, and verification.
- Network performance monitoring:
  - Measurement of jitter, latency, or packet loss in the network.
  - Continuous, reliable, and predictable measurements.
- IP service network health assessment to verify that the existing QoS is sufficient for new IP services.
- Edge-to-edge network availability monitoring for proactive verification and connectivity testing of network resources (for example, shows the network availability of an NFS server used to store business critical data from a remote site).
- Network operation troubleshooting by providing consistent, reliable measurement that immediately identifies problems and saves troubleshooting time.
- Multiprotocol Label Switching (MPLS) performance monitoring and network verification (if the device supports MPLS).

Network Performance Measurement with Cisco IOS IP SLAs

You can use IP SLAs to monitor the performance between any area in the network—core, distribution, and edge—without deploying a physical probe. It uses generated traffic to measure network performance between two networking devices.

Figure 146: Cisco IOS IP SLAs Operation

The following figure shows how IP SLAs begin when the source device sends a generated packet to the destination device. After the destination device receives the packet, depending on the type of IP SLAs operation, it responds with time-stamp information for the source to make the calculation on performance metrics. An IP SLAs operation performs a network measurement from the source device to a destination in the network using a specific protocol such as UDP.



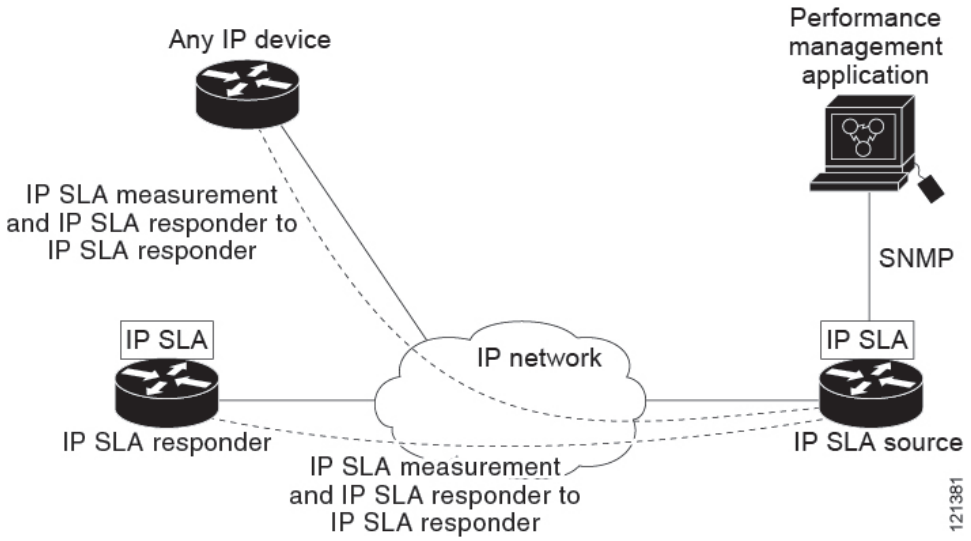
IP SLA Responder and IP SLA Control Protocol

The IP SLA responder is a component embedded in the destination Cisco device that allows the system to anticipate and respond to IP SLA request packets. The responder provides accurate measurements without the need for dedicated probes. The responder uses the Cisco IOS IP SLA Control Protocol to provide a mechanism through which it can be notified on which port it should listen and respond.

Note The IP SLA responder can be a Cisco IOS Layer 2, responder-configurable device. The responder does not need to support full IP SLA functionality.

The following figure shows where the Cisco IOS IP SLA responder fits in the IP network. The responder listens on a specific port for control protocol messages sent by an IP SLA operation. Upon receipt of the control message, it enables the specified UDP or TCP port for the specified duration. During this time, the responder accepts the requests and responds to them. It disables the port after it responds to the IP SLA packet, or when the specified time expires. MD5 authentication for control messages is available for added security.

Figure 147: Cisco IOS IP SLAs Operation



You do not need to enable the responder on the destination device for all IP SLA operations. For example, a responder is not required for services that are already provided by the destination router (such as Telnet or HTTP).

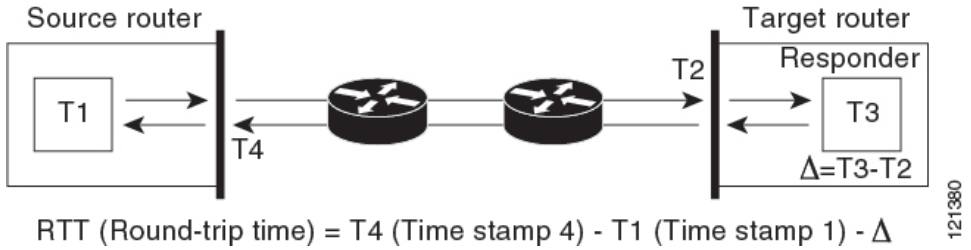
Response Time Computation for IP SLAs

Switches, controllers, and routers can take tens of milliseconds to process incoming packets due to other high priority processes. This delay affects the response times because the test-packet reply might be in a queue while waiting to be processed. In this situation, the response times would not accurately represent true network delays. IP SLAs minimize these processing delays on the source device as well as on the target device (if the responder is being used) to determine true round-trip times. IP SLA test packets use time stamping to minimize the processing delays.

When the IP SLA responder is enabled, it allows the target device to take time stamps when the packet arrives on the interface at interrupt level and again just as it is leaving, eliminating the processing time. This time stamping is made with a granularity of sub-milliseconds (ms).

Figure 148: Cisco IOS IP SLA Responder Time Stamping

The following figure demonstrates how the responder works. Four time stamps are taken to make the calculation for round-trip time. At the target router, with the responder functionality enabled, time stamp 2 (TS2) is subtracted from time stamp 3 (TS3) to produce the time spent processing the test packet as represented by delta. This delta value is then subtracted from the overall round-trip time. Notice that the same principle is applied by IP SLAs on the source router where the incoming time stamp 4 (TS4) is also taken at the interrupt level to allow for greater accuracy.



An additional benefit of the two time stamps at the target device is the ability to track one-way delay, jitter, and directional packet loss. Because much network behavior is asynchronous, it is critical to have these statistics. However, to capture one-way delay measurements, you must configure both the source router and target router with Network Time Protocol (NTP) so that the source and target are synchronized to the same clock source. One-way jitter measurements do not require clock synchronization.

IP SLAs Operation Scheduling

When you configure an IP SLAs operation, you must schedule the operation to begin capturing statistics and collecting error information. You can schedule an operation to start immediately or to start at a certain month, day, and hour. You can use the *pending* option to set the operation to start at a later time. The pending option is an internal state of the operation that is visible through SNMP. The pending state is also used when an operation is a reaction (threshold) operation waiting to be triggered. You can schedule a single IP SLAs operation or a group of operations at one time.

You can schedule several IP SLAs operations by using a single command through the Cisco IOS CLI or the CISCO RTTMON-MIB. Scheduling the operations to run at evenly distributed times allows you to control the amount of IP SLAs monitoring traffic. This distribution of IP SLA operations helps minimize the CPU utilization and thus improves network scalability.

For more details about the IP SLA multi-operations scheduling functionality, see the "IP SLAs—Multiple Operation Scheduling" chapter of the *Cisco IOS IP SLAs Configuration Guide*.

When you try to reconfigure a scheduled IP SLA operation, the active status of the IP SLA operation is checked. If the operation is active, the following message is displayed.

```
Entry already running and cannot be modified.

(only can delete (no) and start over)

(check to see if the probe has finished exiting)
```

You can reconfigure the parameters of a scheduled IP SLA operation. You can use the **configure replace** command to replace the current running configuration with a new configuration. The **configure replace** command is a general command and is not specific to IP SLAs. The command will replace the entire current configuration with the new configuration provided. The IP SLA operation will be stopped, and restarted with the new parameters. You cannot change the IP SLA probe type or IP SLA socket related parameters such as the destination IP address, source IP address, destination port and source port using the **configure replace** command.

IP SLA Operation Threshold Monitoring

To support successful service level agreement monitoring, you must have mechanisms that notify you immediately of any possible violation. IP SLAs can send SNMP traps that are triggered by events such as the following:

- Connection loss
- Timeout
- Round-trip time threshold
- Average jitter threshold
- One-way packet loss

- One-way jitter
- One-way mean opinion score (MOS)
- One-way latency

An IP SLA threshold violation can also trigger another IP SLA operation for further analysis. For example, the frequency could be increased or an Internet Control Message Protocol (ICMP) path echo or ICMP path jitter operation could be initiated for troubleshooting.

ICMP Echo

The ICMP echo operation measures the end-to-end response time between a Cisco device and any other device that uses IP. The response time is computed by measuring the time it takes to send an ICMP echo request message to a destination and receive an ICMP echo reply. Many customers use IP SLA ICMP-based operations, in-house ping testing, or ping-based dedicated probes to measure this response time. The IP SLA ICMP echo operation conforms to the same specifications as ICMP ping testing, and both methods result in the same response times.

UDP Jitter

Jitter is a simple term that describes interpacket delay variance. When multiple packets are sent consecutively at an interval of 10 ms from source to destination, the destination should receive them 10 ms apart (if the network is behaving correctly). However, if there are delays in the network (such as queuing, arriving through alternate routes, and so on), the time interval between packet arrivals might be more or less than 10 ms. A positive jitter value indicates that the packets arrived more than 10 ms apart. A negative jitter value indicates that the packets arrived less than 10 ms apart. If the packets arrive 12 ms apart, the positive jitter is 2 ms; if the packets arrive 8 ms apart, the negative jitter is 2 ms. For delay-sensitive networks, positive jitter values are undesirable, and a jitter value of 0 is ideal.

In addition to monitoring jitter, the IP SLA UDP jitter operation can be used as a multipurpose data gathering operation. The packets generated by IP SLAs carry sequence information and time stamps from the source and operational target that include packet sending and receiving data. Based on this data, UDP jitter operations measure the following:

- Per-direction jitter (source to destination and destination to source)
- Per-direction packet-loss
- Per-direction delay (one-way delay)
- Round-trip delay (average round-trip time)

Because the paths for the sending and receiving of data can be different (asymmetric), you can use the per-direction data to more readily identify where congestion or other problems are occurring in the network.

The UDP jitter operation generates synthetic (simulated) UDP traffic and sends a number of UDP packets, each of a specified size, sent a specified number of milliseconds apart, from a source router to a target router, at a given frequency. By default, ten packet-frames, each with a payload size of 10 bytes are generated every 10 ms, and the operation is repeated every 60 seconds. You can configure each of these parameters to best simulate the IP service you want to provide.

To provide accurate one-way delay (latency) measurements, time synchronization (as provided by NTP) is required between the source and the target device. Time synchronization is not required for the one-way jitter and packet loss measurements. If the time is not synchronized between the source and target devices, one-way

jitter and packet loss data is returned, but values of 0 are returned for the one-way delay measurements provided by the UDP jitter operation.

How to Configure IP SLAs Operations

This section does not include configuration information for all available operations as the configuration information details are included in the *Cisco IOS IP SLAs Configuration Guide*. It does include several operations as examples, including configuring the responder, configuring a UDP jitter operation, which requires a responder, and configuring an ICMP echo operation, which does not require a responder. For details about configuring other operations, see the *Cisco IOS IP SLAs Configuration Guide*.

Default Configuration

No IP SLAs operations are configured.

Configuration Guidelines

For information on the IP SLA commands, see the *Cisco IOS IP SLAs Command Reference, Release 12.4T* .

For detailed descriptions and configuration procedures, see the *Cisco IOS IP SLAs Configuration Guide, Release 12.4TL*.

Not all of the IP SLA commands or operations described in the referenced guide are supported on the device. The device supports IP service level analysis by using UDP jitter, UDP echo, HTTP, TCP connect, ICMP echo, ICMP path echo, ICMP path jitter, FTP, DNS, and DHCP, as well as multiple operation scheduling and proactive threshold monitoring. It does not support VoIP service levels using the gatekeeper registration delay operations measurements.

Before configuring any IP SLAs application, you can use the **show ip sla application** privileged EXEC command to verify that the operation type is supported on your software image. This is an example of the output from the command:

```
Device# show ip sla application

        IP Service Level Agreements
Version: Round Trip Time MIB 2.2.0, Infrastructure Engine-III

Supported Operation Types:
        icmpEcho, path-echo, path-jitter, udpEcho, tcpConnect, http
        dns, udpJitter, dhcp, ftp, udpApp, wspApp

Supported Features:
        IPSLAs Event Publisher

IP SLAs low memory water mark: 33299323
Estimated system max number of entries: 24389

Estimated number of configurable operations: 24389
Number of Entries configured  : 0
Number of active Entries      : 0
Number of pending Entries     : 0
Number of inactive Entries    : 0
Time of last change in whole IP SLAs: *13:04:37.668 UTC Wed Dec 19 2012
```

Configuring the IP SLA Responder

The IP SLA responder is available only on Cisco IOS software-based devices, including some Layer 2 devices that do not support full IP SLA functionality.

Follow these steps to configure the IP SLA responder on the target device (the operational target):

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# config t`	Enters the global configuration mode.
Step 3	**ip sla responder** {**tcp-connect** | **udp-echo**} **ipaddress** *ip-address* **port** *port-number* **Example:** `Device(config)# ip sla responder udp-echo 172.29.139.134 5000`	Configures the device as an IP SLA responder. The keywords have these meanings: • **tcp-connect**—Enables the responder for TCP connect operations. • **udp-echo**—Enables the responder for User Datagram Protocol (UDP) echo or jitter operations. • **ipaddress** *ip-address*—Enter the destination IP address. • **port** *port-number*—Enter the destination port number. **Note** The IP address and port number must match those configured on the source device for the IP SLA operation.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Implementing IP SLA Network Performance Measurement

Follow these steps to implement IP SLA network performance measurement on your device:

Before you begin

Use the **show ip sla application** privileged EXEC command to verify that the desired operation type is supported on your software image.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# config t`	Enters the global configuration mode.
Step 3	**ip sla** *operation-number* **Example:** `Device(config)# ip sla 10`	Creates an IP SLA operation, and enters IP SLA configuration mode.
Step 4	**udp-jitter** {*destination-ip-address* | *destination-hostname*} *destination-port* [**source-ip** {*ip-address* | *hostname*}] [**source-port** *port-number*] [**control** {**enable** | **disable**}] [**num-packets** *number-of-packets*] [**interval** *interpacket-interval*]	Configures the IP SLA operation as the operation type of your choice (a UDP jitter operation is used in the example), and enters its configuration mode (UDP jitter configuration mode is used in the example).

	Command or Action	Purpose
	Example: `Device(config-ip-sla)# udp-jitter 172.29.139.134 5000 source-ip 172.29.139.140 source-port 4000`	• *destination-ip-address* | *destination-hostname*: Specifies the destination IP address or hostname. • *destination-port*: Specifies the destination port number in the range from 1 to 65535. • (Optional) **source-ip** {*ip-address* | *hostname*}: Specifies the source IP address or hostname. When a source IP address or hostname is not specified, IP SLA chooses the IP address nearest to the destination • (Optional) **source-port** *port-number*: Specifies the source port number in the range from 1 to 65535. When a port number is not specified, IP SLA chooses an available port. **Note** If the **udp-jitter** command does not have the source port configured, UDP chooses any random port for control packets. In case UDP chooses the reserved port 1967, it may result in high CPU utilisation by the IP SLA responder. • (Optional) **control**: Enables or disables sending of IP SLA control messages to the IP SLA responder. By default, IP SLA control messages are sent to the destination device to establish a connection with the IP SLA responder • (Optional) **num-packets** *number-of-packets*: Enters the number of packets to be generated. The range is 1 to 6000; the default is 10. • (Optional) **interval** *inter-packet-interval*: Enters the interval between sending packets in milliseconds. The range is 1 to 6000; the default value is 20 ms.
Step 5	**frequency** *seconds* **Example:** `Device(config-ip-sla-jitter)# frequency 45`	(Optional) Configures options for the SLA operation. This example sets the rate at which a specified IP SLA operation repeats. The range is from 1 to 604800 seconds; the default is 60 seconds.

	Command or Action	Purpose
Step 6	**threshold** *milliseconds* **Example:** `Device(config-ip-sla-jitter)# threshold 200`	(Optional) Configures threshold conditions. This example sets the threshold of the specified IP SLA operation to 200. The range is from 0 to 60000 milliseconds.
Step 7	**exit** **Example:** `Device(config-ip-sla-jitter)# exit`	Exits the SLA operation configuration mode (UDP jitter configuration mode in this example), and returns to global configuration mode.
Step 8	**ip sla schedule** *operation-number* [**life** {**forever** | *seconds*}] [**start-time** {*hh:mm* [:*ss*] [*month day* | *day month*] | **pending** | **now** | **after** *hh:mm:ss*] [**ageout** *seconds*] [**recurring**] **Example:** `Device(config)# ip sla schedule 10 start-time now life forever`	Configures the scheduling parameters for an individual IP SLA operation. • *operation-number*: Enter the RTR entry number. • (Optional) **life**: Sets the operation to run indefinitely (**forever**) or for a specific number of *seconds*. The range is from 0 to 2147483647. The default is 3600 seconds (1 hour). • (Optional) **start-time**: Enters the time for the operation to begin collecting information: To start at a specific time, enter the hour, minute, second (in 24-hour notation), and day of the month. If no month is entered, the default is the current month. Enter **pending** to select no information collection until a start time is selected. Enter **now** to start the operation immediately. Enter **after** *hh:mm:ss* to show that the operation should start after the entered time has elapsed. • (Optional) **ageout** *seconds*: Enter the number of seconds to keep the operation in memory when it is not actively collecting information. The range is 0 to 2073600 seconds, the default is 0 seconds (never ages out). • (Optional) **recurring**: Set the operation to automatically run every day.

	Command or Action	Purpose
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 10	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

UDP Jitter Configuration

This example shows how to configure a UDP jitter IP SLA operation:

```
Device(config)# ip sla 10
Device(config-ip-sla)# udp-jitter 172.29.139.134 5000 source-ip 172.29.139.140 source-port
 4000
Device(config-ip-sla-jitter)# frequency 30
Device(config-ip-sla-jitter)# exit
Device(config)# ip sla schedule 10 start-time now life forever
Device(config)# end
Device# show ip sla configuration 10
IP SLAs, Infrastructure Engine-II.

Entry number: 10
Owner:
Tag:
Type of operation to perform: udp-jitter
Target address/Source address: 1.1.1.1/0.0.0.0
Target port/Source port: 2/0
Request size (ARR data portion): 32
Operation timeout (milliseconds): 5000
Packet Interval (milliseconds)/Number of packets: 20/10
Type Of Service parameters: 0x0
Verify data: No
Vrf Name:
Control Packets: enabled
Schedule:
    Operation frequency (seconds): 30
    Next Scheduled Start Time: Pending trigger
    Group Scheduled : FALSE
    Randomly Scheduled : FALSE
    Life (seconds): 3600
    Entry Ageout (seconds): never
    Recurring (Starting Everyday): FALSE
```

```
    Status of entry (SNMP RowStatus): notInService
Threshold (milliseconds): 5000
Distribution Statistics:
    Number of statistic hours kept: 2
    Number of statistic distribution buckets kept: 1
    Statistic distribution interval (milliseconds): 20
Enhanced History:
```

Analyzing IP Service Levels by Using the UDP Jitter Operation

Follow these steps to configure a UDP jitter operation on the source device:

Before you begin

You must enable the IP SLA responder on the target device (the operational target) to configure a UDP jitter operation on the source device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# config t`	Enters the global configuration mode.
Step 3	**ip sla** *operation-number* **Example:** `Device(config)# ip sla 10`	Creates an IP SLA operation, and enters IP SLA configuration mode.
Step 4	**udp-jitter** {*destination-ip-address* | *destination-hostname*} *destination-port* [**source-ip** {*ip-address* | *hostname*}] [**source-port** *port-number*] [**control** {**enable** | **disable**}] [**num-packets** *number-of-packets*] [**interval** *interpacket-interval*] **Example:** `Device(config-ip-sla)# udp-jitter 172.29.139.134 5000 source-ip 172.29.139.140 source-port 4000`	Configures the IP SLA operation as a UDP jitter operation, and enters UDP jitter configuration mode. • *destination-ip-address* | *destination-hostname*: Specifies the destination IP address or hostname. • *destination-port*: Specifies the destination port number in the range from 1 to 65535. • (Optional) **source-ip** {*ip-address* | *hostname*}: Specifies the source IP address or hostname. When a source IP address or hostname is not specified, IP

	Command or Action	Purpose
		SLA chooses the IP address nearest to the destination. • (Optional) **source-port** *port-number*: Specifies the source port number in the range from 1 to 65535. When a port number is not specified, IP SLA chooses an available port. **Note** If the **udp-jitter** command does not have the source port configured, UDP chooses any random port for control packets. In case UDP chooses the reserved port 1967, it may result in high CPU utilisation by the IP SLA responder. • (Optional) **control**: Enables or disables sending of IP SLA control messages to the IP SLA responder. By default, IP SLA control messages are sent to the destination device to establish a connection with the IP SLA responder. • (Optional) **num-packets** *number-of-packets*: Enters the number of packets to be generated. The range is 1 to 6000; the default is 10. • (Optional) **interval** *inter-packet-interval*: Enters the interval between sending packets in milliseconds. The range is 1 to 6000; the default value is 20 ms.
Step 5	**frequency** *seconds* **Example:** `Device(config-ip-sla-jitter)# frequency 45`	(Optional) Sets the rate at which a specified IP SLA operation repeats. The range is from 1 to 604800 seconds; the default is 60 seconds.
Step 6	**exit** **Example:** `Device(config-ip-sla-jitter)# exit`	Exits UDP jitter configuration mode, and returns to global configuration mode.
Step 7	**ip sla schedule** *operation-number* [**life** {**forever** | *seconds*}] [**start-time** {*hh:mm* [*:ss*] [*month day* | *day month*] | **pending** | **now** | **after** *hh:mm:ss*] [**ageout** *seconds*] [**recurring**]	Configures the scheduling parameters for an individual IP SLA operation.

	Command or Action	Purpose
	Example: `Device(config)# ip sla schedule 10 start-time now life forever`	• *operation-number*: Enter the RTR entry number. • (Optional) **life**: Sets the operation to run indefinitely (**forever**) or for a specific number of *seconds*. The range is from 0 to 2147483647. The default is 3600 seconds (1 hour). • (Optional) **start-time**: Enters the time for the operation to begin collecting information: To start at a specific time, enter the hour, minute, second (in 24-hour notation), and day of the month. If no month is entered, the default is the current month. Enter **pending** to select no information collection until a start time is selected. Enter **now** to start the operation immediately. Enter **after** *hh:mm:ss* to show that the operation should start after the entered time has elapsed. • (Optional) **ageout** *seconds*: Enter the number of seconds to keep the operation in memory when it is not actively collecting information. The range is 0 to 2073600 seconds, the default is 0 seconds (never ages out). • (Optional) **recurring**: Set the operation to automatically run every day.
Step 8	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 9	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 10	**copy running-config startup-config** **Example:**	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`Device# copy running-config startup-config`	

Configuring a UDP Jitter IP SLA Operation

This example shows how to configure a UDP jitter IP SLA operation:

```
Device(config)# ip sla 10
Device(config-ip-sla)# udp-jitter 172.29.139.134 5000 source-ip 172.29.139.140 source-port
 4000
Device(config-ip-sla-jitter)# frequency 30
Device(config-ip-sla-jitter)# exit
Device(config)# ip sla schedule 10 start-time now life forever
Device(config)# end
Device# show ip sla configuration 10
IP SLAs, Infrastructure Engine-II.

Entry number: 10
Owner:
Tag:
Type of operation to perform: udp-jitter
Target address/Source address: 1.1.1.1/0.0.0.0
Target port/Source port: 2/0
Request size (ARR data portion): 32
Operation timeout (milliseconds): 5000
Packet Interval (milliseconds)/Number of packets: 20/10
Type Of Service parameters: 0x0
Verify data: No
Vrf Name:
Control Packets: enabled
Schedule:
    Operation frequency (seconds): 30
    Next Scheduled Start Time: Pending trigger
    Group Scheduled : FALSE
    Randomly Scheduled : FALSE
    Life (seconds): 3600
    Entry Ageout (seconds): never
    Recurring (Starting Everyday): FALSE
    Status of entry (SNMP RowStatus): notInService
Threshold (milliseconds): 5000
Distribution Statistics:
    Number of statistic hours kept: 2
    Number of statistic distribution buckets kept: 1
    Statistic distribution interval (milliseconds): 20
Enhanced History:
```

Analyzing IP Service Levels by Using the ICMP Echo Operation

Follow these steps to configure an ICMP echo operation on the source device:

Before you begin

This operation does not require the IP SLA responder to be enabled.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# config terminal`	Enters the global configuration mode.
Step 3	**ip sla** *operation-number* **Example:** `Device(config)# ip sla 10`	Creates an IP SLA operation and enters IP SLA configuration mode.
Step 4	**icmp-echo** {*destination-ip-address* | *destination-hostname*} [**source-ip** {*ip-address* | *hostname*} | **source-interface** *interface-id*] **Example:** `Device(config-ip-sla)# icmp-echo 172.29.139.134`	Configures the IP SLA operation as an ICMP Echo operation and enters ICMP echo configuration mode. • *destination-ip-address* | *destination-hostname*—Specifies the destination IP address or hostname. • (Optional) **source-ip** {*ip-address* | *hostname*}—Specifies the source IP address or hostname. When a source IP address or hostname is not specified, IP SLA chooses the IP address nearest to the destination. • (Optional) **source-interface** *interface-id*—Specifies the source interface for the operation.
Step 5	**frequency** *seconds* **Example:** `Device(config-ip-sla-echo)# frequency 30`	(Optional) Sets the rate at which a specified IP SLA operation repeats. The range is from 1 to 604800 seconds; the default is 60 seconds.
Step 6	**exit** **Example:** `Device(config-ip-sla-echo)# exit`	Exits UDP echo configuration mode, and returns to global configuration mode.

	Command or Action	Purpose
Step 7	**ip sla schedule** *operation-number* [**life** {**forever** | *seconds*}] [**start-time** {*hh:mm* [*:ss*] [*month day* | *day month*] | **pending** | **now** | **after** *hh:mm:ss*] [**ageout** *seconds*] [**recurring**] **Example:** `Device(config)# ip sla schedule 10 start-time now life forever`	Configures the scheduling parameters for an individual IP SLA operation. • *operation-number*—Enter the RTR entry number. • (Optional) **life**—Sets the operation to run indefinitely (**forever**) or for a specific number of *seconds*. The range is from 0 to 2147483647. The default is 3600 seconds (1 hour) • (Optional) **start-time**—Enter the time for the operation to begin collecting information: To start at a specific time, enter the hour, minute, second (in 24-hour notation), and day of the month. If no month is entered, the default is the current month. Enter **pending** to select no information collection until a start time is selected. Enter **now** to start the operation immediately. Enter **after** *hh:mm:ss* to indicate that the operation should start after the entered time has elapsed. • (Optional) **ageout** *seconds*—Enter the number of seconds to keep the operation in memory when it is not actively collecting information. The range is 0 to 2073600 seconds; the default is 0 seconds (never ages out). • (Optional) **recurring**—Sets the operation to automatically run every day.
Step 8	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 9	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.

	Command or Action	Purpose
Step 10	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring an ICMP Echo IP SLA Operation

This example shows how to configure an ICMP echo IP SLA operation:

```
Device(config)# ip sla 10
Device(config-ip-sla)# icmp-echo 172.29.139.134
Device(config-ip-sla-echo)# frequency 30
Device(config-ip-sla-echo)# exit
Device(config)# ip sla schedule 10 start-time now life forever
Device(config)# end
Device# show ip sla configuration 22
IP SLAs, Infrastructure Engine-II.

Entry number: 12
Owner:
Tag:
Type of operation to perform: echo
Target address: 2.2.2.2
Source address: 0.0.0.0
Request size (ARR data portion): 28
Operation timeout (milliseconds): 5000
Type Of Service parameters: 0x0
Verify data: No
Vrf Name:
Schedule:
    Operation frequency (seconds): 60
    Next Scheduled Start Time: Pending trigger
    Group Scheduled : FALSE
    Randomly Scheduled : FALSE
    Life (seconds): 3600
    Entry Ageout (seconds): never
    Recurring (Starting Everyday): FALSE
    Status of entry (SNMP RowStatus): notInService
Threshold (milliseconds): 5000
Distribution Statistics:
    Number of statistic hours kept: 2
    Number of statistic distribution buckets kept: 1
    Statistic distribution interval (milliseconds): 20
History Statistics:
    Number of history Lives kept: 0
    Number of history Buckets kept: 15
    History Filter Type: None
Enhanced History:
```

Monitoring IP SLA Operations

The following table describes the commands used to display IP SLA operation configurations and results:

Table 164: Monitoring IP SLA Operations

Command	Description
show ip sla application	Displays global information about Cisco IOS IP SLAs.
show ip sla authentication	Displays IP SLA authentication information.
show ip sla configuration [*entry-number*]	Displays configuration values including all defaults for all IP SLA operations or a specific operation.
show ip sla enhanced-history {**collection-statistics** | **distribution statistics**} [*entry-number*]	Displays enhanced history statistics for collected history buckets or distribution statistics for all IP SLA operations or a specific operation.
show ip sla ethernet-monitor configuration [*entry-number*]	Displays IP SLA automatic Ethernet configuration.
show ip sla group schedule [*schedule-entry-number*]	Displays IP SLA group scheduling configuration and details.
show ip sla history [*entry-number* | **full** | **tabular**]	Displays history collected for all IP SLA operations.
show ip sla mpls-lsp-monitor {**collection-statistics** | **configuration** | **ldp operational-state** | **scan-queue** | **summary** [*entry-number*] | **neighbors**}	Displays MPLS label switched path (LSP) Health Monitor operations.
show ip sla reaction-configuration [*entry-number*]	Displays the configured proactive threshold monitoring settings for all IP SLA operations or a specific operation.
show ip sla reaction-trigger [*entry-number*]	Displays the reaction trigger information for all IP SLA operations or a specific operation.
show ip sla responder	Displays information about the IP SLA responder.
show ip sla statistics [*entry-number* | **aggregated** | **details**]	Displays current or aggregated operational status and statistics.

Monitoring IP SLA Operation Examples

The following example shows all IP SLAs by application:

```
Device# show ip sla application

        IP Service Level Agreements
Version: Round Trip Time MIB 2.2.0, Infrastructure Engine-III
```

```
Supported Operation Types:
        icmpEcho, path-echo, path-jitter, udpEcho, tcpConnect, http
        dns, udpJitter, dhcp, ftp, udpApp, wspApp

Supported Features:
        IPSLAs Event Publisher

IP SLAs low memory water mark: 33299323
Estimated system max number of entries: 24389

Estimated number of configurable operations: 24389
Number of Entries configured  : 0
Number of active Entries      : 0
Number of pending Entries     : 0
Number of inactive Entries    : 0
Time of last change in whole IP SLAs: *13:04:37.668 UTC Wed Dec 19 2012
```

The following example shows all IP SLA distribution statistics:

```
Device# show ip sla enhanced-history distribution-statistics

Point by point Enhanced History
Entry    = Entry Number
Int      = Aggregation Interval
BucI     = Bucket Index
StartT   = Aggregation Start Time
Pth      = Path index
Hop      = Hop in path index
Comps    = Operations completed
OvrTh    = Operations completed over thresholds
SumCmp   = Sum of RTT (milliseconds)
SumCmp2L = Sum of RTT squared low 32 bits (milliseconds)
SumCmp2H = Sum of RTT squared high 32 bits (milliseconds)
TMax     = RTT maximum (milliseconds)
TMin     = RTT minimum (milliseconds)

Entry Int BucI StartT     Pth Hop Comps OvrTh SumCmp     SumCmp2L   SumCmp2H   T
Max    TMin
```

CHAPTER 163

Configuring SPAN and RSPAN

Prerequisites for SPAN and RSPAN

SPAN

You can limit SPAN traffic to specific VLANs by using the **filter vlan** keyword. If a trunk port is being monitored, only traffic on the VLANs specified with this keyword is monitored.

Note By default, all VLANs are monitored on a trunk port.

RSPAN

You can configure an RSPAN source or a destination session.

Note We recommend that you configure an RSPAN VLAN before you configure an RSPAN source or a destination session.

Restrictions for SPAN and RSPAN

SPAN

The restrictions for SPAN are as follows:

- On each device, you can use 1 to 66 span session number for configuration of SPAN or RSPAN. A maximum of 8 monitor sessions can be configured in a mix of SPAN and RSPAN.
- For SPAN sources, you can monitor traffic for a single port or VLAN or a series or range of ports or VLANs for each session. You cannot mix source ports and source VLANs within a single SPAN session.
- The destination port cannot be a source port; a source port cannot be a destination port.
- You cannot have two SPAN sessions using the same destination port.
- When you configure a device port as a SPAN destination port, it is no longer a normal device port; only monitored traffic passes through the SPAN destination port.
- Entering SPAN configuration commands does not remove previously configured SPAN parameters. You must enter the **no monitor session** {*session_number* | **all** | **local** | **remote**} global configuration command to delete configured SPAN parameters.
- For local SPAN, outgoing packets through the SPAN destination port carry the original encapsulation headers—untagged, ISL, or IEEE 802.1Q—if the **encapsulation replicate** keywords are specified. If the keywords are not specified, the packets are sent in native form.
- You can configure a disabled port to be a source or destination port, but the SPAN function does not start until the destination port and at least one source port or source VLAN are enabled.
- You cannot mix source VLANs and filter VLANs within a single SPAN session.

Traffic monitoring in a SPAN session has the following restrictions:

- Sources can be ports or VLANs, but you cannot mix source ports and source VLANs in the same session.
- Wireshark does not capture egress packets when egress span is active.
- You can run both a local SPAN and an RSPAN source session in the same device. The device supports a total of 66 source and RSPAN destination sessions.
- You can configure two separate SPAN or RSPAN source sessions with separate or overlapping sets of SPAN source ports and VLANs. Both switched and routed ports can be configured as SPAN sources and destinations.
- You can have multiple destination ports in a SPAN session, but no more than 64 destination ports per device. .
- SPAN sessions do not interfere with the normal operation of the device. However, an oversubscribed SPAN destination, for example, a 10-Mb/s port monitoring a 100-Mb/s port, can result in dropped or lost packets.
- When SPAN or RSPAN is enabled, each packet being monitored is sent twice, once as normal traffic and once as a monitored packet. Monitoring a large number of ports or VLANs could potentially generate large amounts of network traffic.
- You can configure SPAN sessions on disabled ports; however, a SPAN session does not become active unless you enable the destination port and at least one source port or VLAN for that session.
- The device does not support a combination of local SPAN and RSPAN in a single session.
  - An RSPAN source session cannot have a local destination port.
  - An RSPAN destination session cannot have a local source port.

- An RSPAN destination session and an RSPAN source session that are using the same RSPAN VLAN cannot run on the same device.

RSPAN

The restrictions for RSPAN are as follows:

- RSPAN does not support BPDU packet monitoring or other Layer 2 device protocols.
- The RSPAN VLAN is configured only on trunk ports and not on access ports. To avoid unwanted traffic in RSPAN VLANs, make sure that the VLAN remote-span feature is supported in all the participating devices.
- RSPAN VLANs are included as sources for port-based RSPAN sessions when source trunk ports have active RSPAN VLANs. RSPAN VLANs can also be sources in SPAN sessions. However, since the device does not monitor spanned traffic, it does not support egress spanning of packets on any RSPAN VLAN identified as the destination of an RSPAN source session on the device.
- If you enable VTP and VTP pruning, RSPAN traffic is pruned in the trunks to prevent the unwanted flooding of RSPAN traffic across the network for VLAN IDs that are lower than 1005.
- It is recommended not to configure RSPAN VLAN as Native VLAN.
- RSPAN sessions do not capture DHCP-inject packets. DHCP-inject packets refer to DHCP packets (DISCOVER, OFFER, REQUEST, and ACK packets) which are modified by the CPU and inserted back into the network.

Information About SPAN and RSPAN

The following sections provide information about SPAN and RSPAN.

SPAN and RSPAN

You can analyze network traffic passing through ports or VLANs by using SPAN or RSPAN to send a copy of the traffic to another port on the device or on another device that has been connected to a network analyzer or other monitoring or security device. SPAN copies (or mirrors) traffic received or sent (or both) on source ports or source VLANs to a destination port for analysis. SPAN does not affect the switching of network traffic on the source ports or VLANs. You must dedicate the destination port for SPAN use. Except for traffic that is required for the SPAN or RSPAN session, destination ports do not receive or forward traffic.

Only traffic that enters or leaves source ports or traffic that enters or leaves source VLANs can be monitored by using SPAN; traffic routed to a source VLAN cannot be monitored. For example, if incoming traffic is being monitored, traffic that gets routed from another VLAN to the source VLAN cannot be monitored; however, traffic that is received on the source VLAN and routed to another VLAN can be monitored.

You can use the SPAN or RSPAN destination port to inject traffic from a network security device. For example, if you connect a Cisco Intrusion Detection System (IDS) sensor appliance to a destination port, the IDS device can send TCP reset packets to close down the TCP session of a suspected attacker.

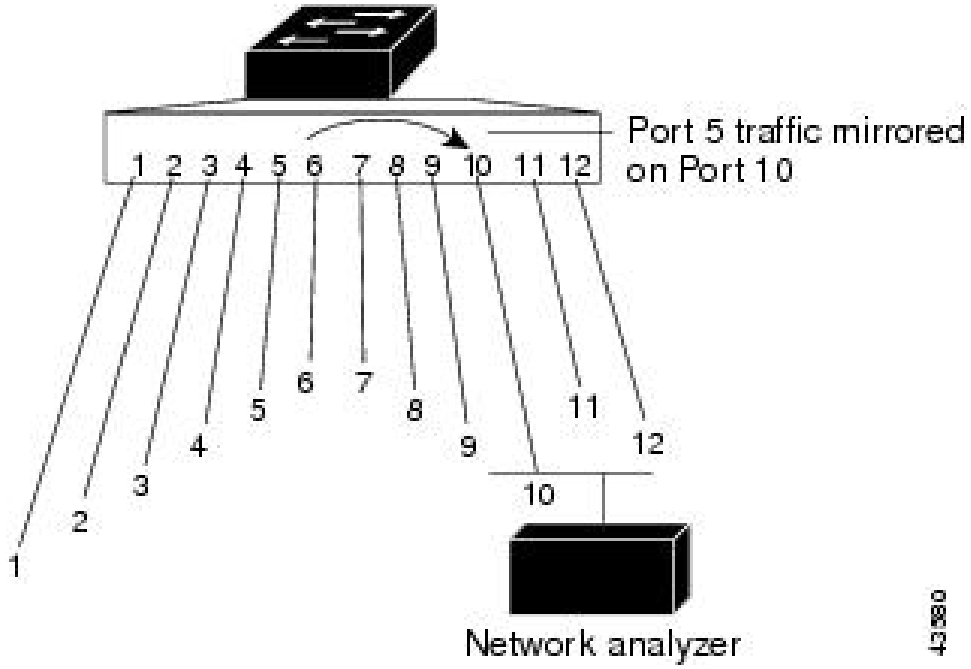
Local SPAN

Local SPAN supports a SPAN session entirely within one device; all source ports or source VLANs and destination ports are in the same device. Local SPAN copies traffic from one or more source ports in any VLAN or from one or more VLANs to a destination port for analysis.

Local SPAN supports a SPAN session entirely within one switch; all source ports and destination ports are in the same switch. Local SPAN copies traffic from one or more source ports to a destination port for analysis.

Figure 149: Example of Local SPAN Configuration on a Single Device

All traffic on port 5 (the source port) is mirrored to port 10 (the destination port). A network analyzer on port 10 receives all network traffic from port 5 without being physically attached to port 5.



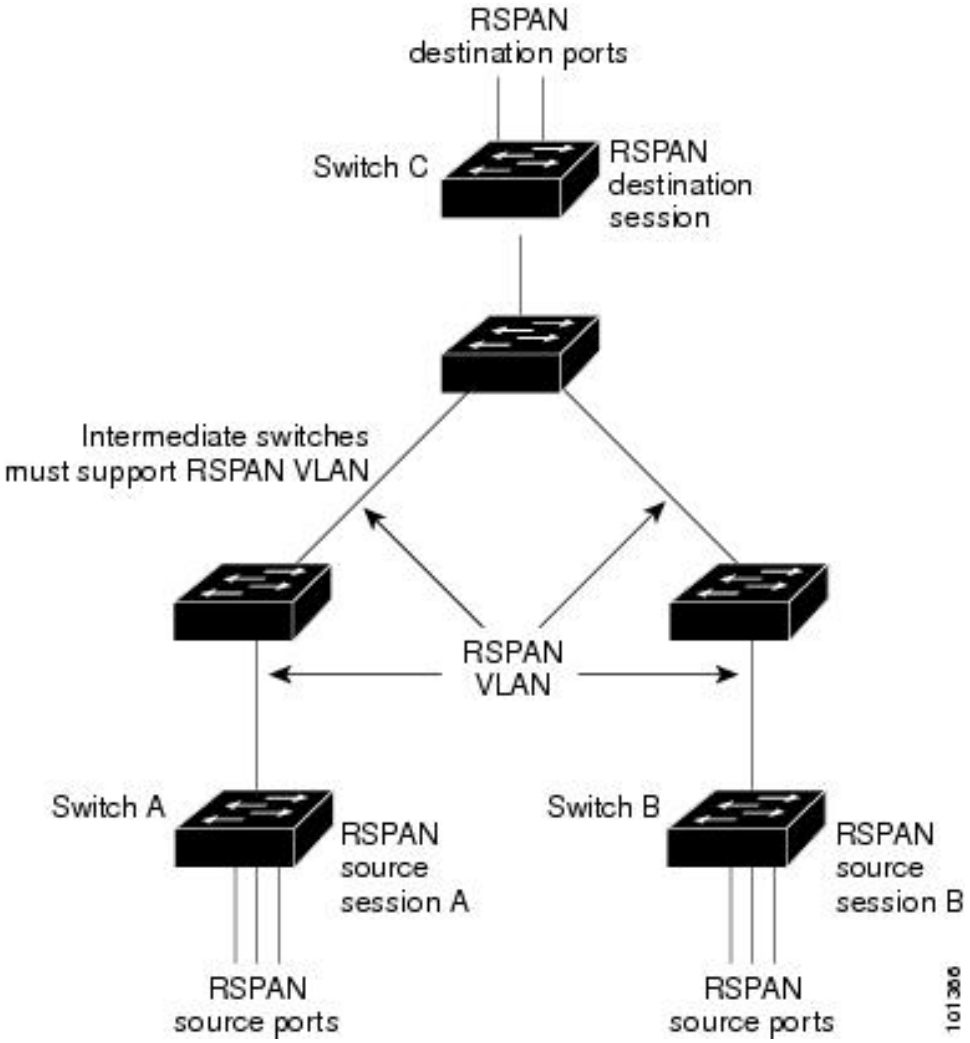
Remote SPAN

RSPAN supports source ports, source VLANs, and destination ports on different devices, enabling remote monitoring of multiple devices across your network.

Figure 150: Example of RSPAN Configuration

The figure below shows source ports on Device A and Device B. The traffic for each RSPAN session is carried over a user-specified RSPAN VLAN that is dedicated for that RSPAN session in all participating devices. The RSPAN traffic from the source ports or VLANs is copied into the RSPAN VLAN and forwarded over trunk ports carrying the RSPAN VLAN to a destination session monitoring the RSPAN VLAN. Each RSPAN source device must have either ports or VLANs as RSPAN sources. The destination is always a physical port,

as shown on Device C in the figure.



SPAN and RSPAN Concepts and Terminology

SPAN Sessions

SPAN sessions (local or remote) allow you to monitor traffic on one or more ports, or one or more VLANs, and send the monitored traffic to one or more destination ports.

A local SPAN session is an association of a destination port with source ports or source VLANs, all on a single network device. Local SPAN does not have separate source and destination sessions. Local SPAN sessions gather a set of ingress and egress packets specified by the user and form them into a stream of SPAN data, which is directed to the destination port.

RSPAN consists of at least one RSPAN source session, an RSPAN VLAN, and at least one RSPAN destination session. You separately configure RSPAN source sessions and RSPAN destination sessions on different network devices. To configure an RSPAN source session on a device, you associate a set of source ports or source VLANs with an RSPAN VLAN. The output of this session is the stream of SPAN packets that are sent to the RSPAN VLAN. To configure an RSPAN destination session on another device, you associate the

destination port with the RSPAN VLAN. The destination session collects all RSPAN VLAN traffic and sends it out the RSPAN destination port.

An RSPAN source session is very similar to a local SPAN session, except for where the packet stream is directed. In an RSPAN source session, SPAN packets are relabeled with the RSPAN VLAN ID and directed over normal trunk ports to the destination device.

An RSPAN destination session takes all packets received on the RSPAN VLAN, strips off the VLAN tagging, and presents them on the destination port. The session presents a copy of all RSPAN VLAN packets (except Layer 2 control packets) to the user for analysis.

Traffic monitoring in a SPAN session has these restrictions:

- Sources can be ports or VLANs, but you cannot mix source ports and source VLANs in the same session.
- You can run both a local SPAN and an RSPAN source session in the same device. The device supports a total of 66 source and RSPAN destination sessions.
- You can configure two separate SPAN or RSPAN source sessions with separate or overlapping sets of SPAN source ports and VLANs. Both switched and routed ports can be configured as SPAN sources and destinations.
- SPAN sessions do not interfere with the normal operation of the device. However, an oversubscribed SPAN destination, for example, a 10-Mb/s port monitoring a 100-Mb/s port, can result in dropped or lost packets.
- When SPAN or RSPAN is enabled, each packet being monitored is sent twice, once as normal traffic and once as a monitored packet. Therefore monitoring a large number of ports or VLANs could potentially generate large amounts of network traffic.
- You can configure SPAN sessions on disabled ports; however, a SPAN session does not become active unless you enable the destination port and at least one source port or VLAN for that session.
- The device does not support a combination of local SPAN and RSPAN in a single session.
  - An RSPAN source session cannot have a local destination port.
  - An RSPAN destination session cannot have a local source port.
  - An RSPAN destination session and an RSPAN source session that are using the same RSPAN VLAN cannot run on the same device.

Monitored Traffic

SPAN sessions can monitor these traffic types:

- Receive (Rx) SPAN—Receive (or ingress) SPAN monitors as much as possible all of the packets received by the source interface or VLAN before any modification or processing is performed by the device. A copy of each packet received by the source is sent to the destination port for that SPAN session.

  Packets that are modified because of routing or Quality of Service (QoS)—for example, modified Differentiated Services Code Point (DSCP)—are copied before modification.

  Features that can cause a packet to be dropped during receive processing have no effect on ingress SPAN; the destination port receives a copy of the packet even if the actual incoming packet is dropped. These features include IP standard and extended input Access Control Lists (ACLs), ingress QoS policing, VLAN ACLs, and egress QoS policing.

- Transmit (Tx) SPAN—Transmit (or egress) SPAN monitors as much as possible all of the packets sent by the source interface after all modification and processing is performed by the device. A copy of each packet sent by the source is sent to the destination port for that SPAN session. The copy is provided after the packet is modified.

  Packets that are modified because of routing (for example, with modified time-to-live (TTL), MAC address, or QoS values) are duplicated (with the modifications) at the destination port.

  Features that can cause a packet to be dropped during transmit processing also affect the duplicated copy for SPAN. These features include IP standard and extended output ACLs and egress QoS policing.

- Both—In a SPAN session, you can also monitor a port or VLAN for both received and sent packets. This is the default.

Therefore, a local SPAN session with encapsulation replicate enabled can have a mixture of untagged and IEEE 802.1Q tagged packets appear on the destination port.

Device congestion can cause packets to be dropped at ingress source ports, egress source ports, or SPAN destination ports. In general, these characteristics are independent of one another. For example:

- A packet might be forwarded normally but dropped from monitoring due to an oversubscribed SPAN destination port.
- An ingress packet might be dropped from normal forwarding, but still appear on the SPAN destination port.
- An egress packet dropped because of device congestion is also dropped from egress SPAN.

In some SPAN configurations, multiple copies of the same source packet are sent to the SPAN destination port. For example, a bidirectional (both Rx and Tx) SPAN session is configured for the Rx monitor on port A and Tx monitor on port B. If a packet enters the device through port A and is switched to port B, both incoming and outgoing packets are sent to the destination port. Both packets are the same unless a Layer 3 rewrite occurs, in which case the packets are different because of the packet modification.

Source Ports

A source port (also called a monitored port) is a switched or routed port that you monitor for network traffic analysis.

In a local SPAN session or RSPAN source session, you can monitor source ports or VLANs for traffic in one or both directions.

The device supports any number of source ports (up to the maximum number of available ports on the device) and any number of source VLANs (up to the maximum number of VLANs supported).

You cannot mix ports and VLANs in a single session.

A source port has these characteristics:

- It can be monitored in multiple SPAN sessions.
- Each source port can be configured with a direction (ingress, egress, or both) to monitor.
- It can be any port type (for example, EtherChannel, Gigabit Ethernet, and so forth).
- For EtherChannel sources, you can monitor traffic for the entire EtherChannel or individually on a physical port as it participates in the port channel.
- It can be an access port, trunk port, routed port, or voice VLAN port.

- It cannot be a destination port.
- Source ports can be in the same or different VLANs.
- You can monitor multiple source ports in a single session.

Source VLANs

VLAN-based SPAN (VSPAN) is the monitoring of the network traffic in one or more VLANs. The SPAN or RSPAN source interface in VSPAN is a VLAN ID, and traffic is monitored on all the ports for that VLAN.

VSPAN has these characteristics:

- All active ports in the source VLAN are included as source ports and can be monitored in either or both directions.
- On a given port, only traffic on the monitored VLAN is sent to the destination port.
- If a destination port belongs to a source VLAN, it is excluded from the source list and is not monitored.
- If ports are added to or removed from the source VLANs, the traffic on the source VLAN received by those ports is added to or removed from the sources being monitored.
- You cannot use filter VLANs in the same session with VLAN sources.
- You can monitor only Ethernet VLANs.

VLAN Filtering

When you monitor a trunk port as a source port, by default, all VLANs active on the trunk are monitored. You can limit SPAN traffic monitoring on trunk source ports to specific VLANs by using VLAN filtering.

- VLAN filtering applies only to trunk ports or to voice VLAN ports.
- VLAN filtering applies only to port-based sessions and is not allowed in sessions with VLAN sources.
- When a VLAN filter list is specified, only those VLANs in the list are monitored on trunk ports or on voice VLAN access ports.
- SPAN traffic coming from other port types is not affected by VLAN filtering; that is, all VLANs are allowed on other ports.
- VLAN filtering affects only traffic forwarded to the destination SPAN port and does not affect the switching of normal traffic.

Destination Port

Each local SPAN session or RSPAN destination session must have a destination port (also called a monitoring port) that receives a copy of traffic from the source ports or VLANs and sends the SPAN packets to the user, usually a network analyzer.

A destination port has these characteristics:

- For a local SPAN session, the destination port must reside on the same device as the source port. For an RSPAN session, it is located on the device containing the RSPAN destination session. There is no destination port on a device running only an RSPAN source session.

- When a port is configured as a SPAN destination port, the configuration overwrites the original port configuration. When the SPAN destination configuration is removed, the port reverts to its previous configuration. If a configuration change is made to the port while it is acting as a SPAN destination port, the change does not take effect until the SPAN destination configuration had been removed.
- If the port was in an EtherChannel group, it is removed from the group while it is a destination port. If it was a routed port, it is no longer a routed port.
- It can be any Ethernet physical port.
- It cannot be a secure port.
- It cannot be a source port.
- It can be an EtherChannel group (**ON** mode only).
- It can participate in only one SPAN session at a time (a destination port in one SPAN session cannot be a destination port for a second SPAN session).
- When it is active, incoming traffic is disabled. The port does not transmit any traffic except that required for the SPAN session. Incoming traffic is never learned or forwarded on a destination port.
- If ingress traffic forwarding is enabled for a network security device, the destination port forwards traffic at Layer 2.
- It does not participate in any of the Layer 2 protocols (STP, VTP, CDP, DTP, PagP).
- A destination port that belongs to a source VLAN of any SPAN session is excluded from the source list and is not monitored.
- The maximum number of destination ports in a device is 64.

Local SPAN and RSPAN destination ports function differently with VLAN tagging and encapsulation:

- For local SPAN, if the **encapsulation replicate** keywords are specified for the destination port, these packets appear with the original encapsulation (untagged, ISL, or IEEE 802.1Q). If these keywords are not specified, packets appear in the untagged format. Therefore, the output of a local SPAN session with **encapsulation replicate** enabled can contain a mixture of untagged, ISL, or IEEE 802.1Q-tagged packets.
- For RSPAN, the original VLAN ID is lost because it is overwritten by the RSPAN VLAN identification. Therefore, all packets appear on the destination port as untagged.

RSPAN VLAN

The RSPAN VLAN carries SPAN traffic between RSPAN source and destination sessions. RSPAN VLAN has these special characteristics:

- All traffic in the RSPAN VLAN is always flooded.
- No MAC address learning occurs on the RSPAN VLAN.
- RSPAN VLAN traffic only flows on trunk ports.
- RSPAN VLANs must be configured in VLAN configuration mode by using the **remote-span** VLAN configuration mode command.
- STP can run on RSPAN VLAN trunks but not on SPAN destination ports.
- An RSPAN VLAN cannot be a private-VLAN primary or secondary VLAN.

For VLANs 1 to 1005 that are visible to VLAN Trunking Protocol (VTP), the VLAN ID and its associated RSPAN characteristic are propagated by VTP. If you assign an RSPAN VLAN ID in the extended VLAN range (1006 to 4094), you must manually configure all intermediate devices.

It is normal to have multiple RSPAN VLANs in a network at the same time with each RSPAN VLAN defining a network-wide RSPAN session. That is, multiple RSPAN source sessions anywhere in the network can contribute packets to the RSPAN session. It is also possible to have multiple RSPAN destination sessions throughout the network, monitoring the same RSPAN VLAN and presenting traffic to the user. The RSPAN VLAN ID separates the sessions.

SPAN and RSPAN Interaction with Other Features

SPAN interacts with these features:

- Routing—SPAN does not monitor routed traffic. VSPAN only monitors traffic that enters or exits the device, not traffic that is routed between VLANs. For example, if a VLAN is being Rx-monitored and the device routes traffic from another VLAN to the monitored VLAN, that traffic is not monitored and not received on the SPAN destination port.
- STP—A destination port does not participate in STP while its SPAN or RSPAN session is active. The destination port can participate in STP after the SPAN or RSPAN session is disabled. On a source port, SPAN does not affect the STP status. STP can be active on trunk ports carrying an RSPAN VLAN.
- CDP—A SPAN destination port does not participate in CDP while the SPAN session is active. After the SPAN session is disabled, the port again participates in CDP.
- VTP—You can use VTP to prune an RSPAN VLAN between devices.
- VLAN and trunking—You can modify VLAN membership or trunk settings for source or destination ports at any time. However, changes in VLAN membership or trunk settings for a destination port do not take effect until you remove the SPAN destination configuration. Changes in VLAN membership or trunk settings for a source port immediately take effect, and the respective SPAN sessions automatically adjust accordingly.
- EtherChannel—You can configure an EtherChannel group as a source port. When a group is configured as a SPAN source, the entire group is monitored.

  If a physical port is added to a monitored EtherChannel group, the new port is added to the SPAN source port list. If a port is removed from a monitored EtherChannel group, it is automatically removed from the source port list.

  A physical port that belongs to an EtherChannel group can be configured as a SPAN source port and still be a part of the EtherChannel. In this case, data from the physical port is monitored as it participates in the EtherChannel. However, if a physical port that belongs to an EtherChannel group is configured as a SPAN destination, it is removed from the group. After the port is removed from the SPAN session, it rejoins the EtherChannel group. Ports removed from an EtherChannel group remain members of the group, but they are in the inactive or suspended state.

  If a physical port that belongs to an EtherChannel group is a destination port and the EtherChannel group is a source, the port is removed from the EtherChannel group and from the list of monitored ports.
- Multicast traffic can be monitored. For egress and ingress port monitoring, only a single unedited packet is sent to the SPAN destination port. It does not reflect the number of times the multicast packet is sent.
- A private-VLAN port cannot be a SPAN destination port.
- A secure port cannot be a SPAN destination port.

For SPAN sessions, do not enable port security on ports with monitored egress when ingress forwarding is enabled on the destination port. For RSPAN source sessions, do not enable port security on any ports with monitored egress.

- An IEEE 802.1x port can be a SPAN source port. You can enable IEEE 802.1x on a port that is a SPAN destination port; however, IEEE 802.1x is disabled until the port is removed as a SPAN destination.

  For SPAN sessions, do not enable IEEE 802.1x on ports with monitored egress when ingress forwarding is enabled on the destination port. For RSPAN source sessions, do not enable IEEE 802.1x on any ports that are egress monitored.

- DHCP Snooping and Local SPAN can be configured on the same VLAN for non-SDA deployments.

Flow-Based SPAN

You can control the type of network traffic to be monitored in SPAN or RSPAN sessions by using flow-based SPAN (FSPAN) or flow-based RSPAN (FRSPAN), which apply access control lists (ACLs) to the monitored traffic on the source ports. The FSPAN ACLs can be configured to filter IPv4, IPv6, and non-IP monitored traffic.

You apply an ACL to a SPAN session through the interface. It is applied to all the traffic that is monitored on all interfaces in the SPAN session.The packets that are permitted by this ACL are copied to the SPAN destination port. No other packets are copied to the SPAN destination port.

The original traffic continues to be forwarded, and any port, VLAN, and router ACLs attached are applied. The FSPAN ACL does not have any effect on the forwarding decisions. Similarly, the port, VLAN, and router ACLs do not have any effect on the traffic monitoring. If a security input ACL denies a packet and it is not forwarded, the packet is still copied to the SPAN destination ports if the FSPAN ACL permits it. But if the security output ACL denies a packet and it is not sent, it is not copied to the SPAN destination ports. However, if the security output ACL permits the packet to go out, it is only copied to the SPAN destination ports if the FSPAN ACL permits it. This is also true for an RSPAN session.

You can attach three types of FSPAN ACLs to the SPAN session:

- IPv4 FSPAN ACL— Filters only IPv4 packets.
- IPv6 FSPAN ACL— Filters only IPv6 packets.
- MAC FSPAN ACL— Filters only non-IP packets.

The FSPAN ACL continues to be correctly applied, and traffic is copied to the SPAN destination ports on the devices where the FSPAN ACL fits in the hardware memory.

When an empty FSPAN ACL is attached, some hardware functions copy all traffic to the SPAN destination ports for that ACL. If sufficient hardware resources are not available, even an empty FSPAN ACL can be unloaded.

Default SPAN and RSPAN Configuration

Table 165: Default SPAN and RSPAN Configuration

Feature	Default Setting
SPAN state (SPAN and RSPAN)	Disabled.
Source port traffic to monitor	Both received and sent traffic (**both**).

Feature	Default Setting
Encapsulation type (destination port)	Native form (untagged packets).
Ingress forwarding (destination port)	Disabled.
VLAN filtering	On a trunk interface used as a source port, all VLANs are monitored.
RSPAN VLANs	None configured.

Configuring SPAN and RSPAN

SPAN Configuration Guidelines

- To remove a source or destination port or VLAN from the SPAN session, use the **no monitor session** *session_number* **source interface** *interface-id* {**interface** *interface-id* | **vlan** *vlan-id*} global configuration command or the **no monitor session** *session_number* **destination interface** *interface-id* global configuration command. For destination interfaces, the **encapsulation** options are ignored with the **no** form of the command.
- To monitor all VLANs on the trunk port, use the **no monitor session** *session_number* **filter** global configuration command.

RSPAN Configuration Guidelines

- All the SPAN configuration guidelines apply to RSPAN.
- As RSPAN VLANs have special properties, you should reserve a few VLANs across your network for use as RSPAN VLANs; do not assign access ports to these VLANs.
- You can apply an output ACL to RSPAN traffic to selectively filter or monitor specific packets. Specify these ACLs on the RSPAN VLAN in the RSPAN source device.
- For RSPAN configuration, you can distribute the source ports and the destination ports across multiple devices in your network.
- Access ports (including voice VLAN ports) on the RSPAN VLAN are put in the inactive state.
- You can configure any VLAN as an RSPAN VLAN as long as these conditions are met:
  - The same RSPAN VLAN is used for an RSPAN session in all the devices.
  - All participating devices support RSPAN.

FSPAN and FRSPAN Configuration Guidelines

- When at least one FSPAN ACL is attached, FSPAN is enabled.

- When you attach at least one FSPAN ACL that is not empty to a SPAN session, and you have not attached one or more of the other FSPAN ACLs (for instance, you have attached an IPv4 ACL that is not empty, and have not attached IPv6 and MAC ACLs), FSPAN blocks the traffic that would have been filtered by the unattached ACLs. Therefore, this traffic is not monitored.

How to Configure SPAN and RSPAN

The following sections provide information on how to configure SPAN and RSPAN.

Creating a Local SPAN Session

Follow these steps to create a SPAN session and specify the source (monitored) ports or VLANs and the destination (monitoring) ports.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session all`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source** {**interface** *interface-id* | **vlan** *vlan-id*} [**,** | **-**] [**both** | **rx** | **tx**] **Example:** `Device(config)# monitor session 1 source interface gigabitethernet1/1`	Specifies the SPAN session and the source port (monitored port). • For *session_number*, the range is 1 to 66. • For *interface-id*, specify the source port to monitor. Valid interfaces include physical interfaces and port-channel logical interfaces (**port-channel**

	Command or Action	Purpose
		port-channel-number). Valid port-channel numbers are 1 to 48. • For *vlan-id*, specify the source VLAN to monitor. The range is 1 to 4094 (excluding the RSPAN VLAN). **Note** A single session can include multiple sources (ports or VLANs) defined in a series of commands, but you cannot combine source ports and source VLANs in one session. • (Optional) [**,** | **-**] Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. • (Optional) **both** | **rx** | **tx**—Specifies the direction of traffic to monitor. If you do not specify a traffic direction, the source interface sends both sent and received traffic. • **both**—Monitors both received and sent traffic. • **rx**—Monitors received traffic. • **tx**—Monitors sent traffic. **Note** You can use the **monitor session** *session_number* **source** command multiple times to configure multiple source ports.
Step 5	**monitor session** *session_number* **destination** {**interface** *interface-id* [**,** | **-**] [**encapsulation {replicate | dot1q}**]} **Example:** `Device(config)# monitor session 1 destination interface gigabitethernet 1/1 encapsulation replicate`	Specifies the SPAN session and the destination port (monitoring port). The port LED changes to amber when the configuration changes take effect. The LED returns to its original state (green) only after removing the SPAN destination configuration. **Note** For local SPAN, you must use the same session number for the source and destination interfaces. • For *session_number*, specify the session number entered in step 4.

	Command or Action	Purpose
		• For *interface-id*, specify the destination port. The destination interface must be a physical port; it cannot be an EtherChannel, and it cannot be a VLAN. • (Optional) [**,** | **-**] Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. (Optional) **encapsulation replicate** specifies that the destination interface replicates the source interface encapsulation method. If not selected, the default is to send packets in native form (untagged). (Optional) **encapsulation dot1q**specifies that the destination interface accepts the source interface incoming packets with IEEE 802.1Q encapsulation. **Note** You can use **monitor session** *session_number* **destination** command multiple times to configure multiple destination ports.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Creating a Local SPAN Session and Configuring Incoming Traffic

Follow these steps to create a SPAN session, to specify the source ports or VLANs and the destination ports, and to enable incoming traffic on the destination port for a network security device (such as a Cisco IDS Sensor Appliance).

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session all`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source** {**interface** *interface-id* | **vlan** *vlan-id*} [**,** | **-**] [**both** | **rx** | **tx**] **Example:** `Device(config)# monitor session 2 source gigabitethernet1/1 rx`	Specifies the SPAN session and the source port (monitored port).
Step 5	**monitor session** *session_number* **destination** {**interface** *interface-id* [**,** | **-**] [**encapsulation replicate**] [**ingress** {**dot1q vlan** *vlan-id* | **untagged vlan** *vlan-id* | **vlan** *vlan-id*}]} **Example:** `Device(config)# monitor session 2 destination interface gigabitethernet1/2 encapsulation replicate ingress dot1q vlan 6`	Specifies the SPAN session, the destination port, the packet encapsulation, and the ingress VLAN and encapsulation. • For *session_number*, specify the session number entered in Step 4. • For *interface-id*, specify the destination port. The destination interface must be a physical port; it cannot be an EtherChannel, and it cannot be a VLAN.

	Command or Action	Purpose
		• (Optional) [**,** | -]—Specifies a series or range of interfaces. Enter a space before and after the comma or hyphen. • (Optional) **encapsulation replicate** specifies that the destination interface replicates the source interface encapsulation method. If not selected, the default is to send packets in native form (untagged). • (Optional) **encapsulation dot1q**specifies that the destination interface accepts the source interface incoming packets with IEEE 802.1Q encapsulation. • **ingress** enables forwarding of incoming traffic on the destination port and to specify the encapsulation type: • **dot1q vlan** *vlan-id*—Accepts incoming packets with IEEE 802.1Q encapsulation with the specified VLAN as the default VLAN. • **untagged vlan** *vlan-id* or **vlan** *vlan-id*—Accepts incoming packets with untagged encapsulation type with the specified VLAN as the default VLAN.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Specifying VLANs to Filter

Follow these steps to limit SPAN source traffic to specific VLANs.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session all`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source interface** *interface-id* **Example:** `Device(config)# monitor session 2 source interface gigabitethernet 1/1 rx`	Specifies the characteristics of the source port (monitored port) and SPAN session. • For *session_number*, the range is 1 to 66. • For *interface-id*, specify the source port to monitor. The interface specified must already be configured as a trunk port.
Step 5	**monitor session** *session_number* **filter vlan** *vlan-id* [**,** | **-**] **Example:** `Device(config)# monitor session 2 filter vlan 1 - 5 , 9`	Limits the SPAN source traffic to specific VLANs. • For *session_number*, enter the session number specified in Step 4. • For *vlan-id*, the range is 1 to 4094. • (Optional) Use a comma (**,**) to specify a series of VLANs, or use a hyphen (**-**) to specify a range of VLANs. Enter a space before and after the comma; enter a space before and after the hyphen.

	Command or Action	Purpose
Step 6	**monitor session** *session_number* **destination** {**interface** *interface-id* [**,** | -] [**encapsulation replicate**]} **Example:** `Device(config)# monitor session 2 destination interface gigabitethernet 1/1`	Specifies the SPAN session and the destination port (monitoring port). • For *session_number*, specify the session number entered in Step 4. • For *interface-id*, specify the destination port. The destination interface must be a physical port; it cannot be an EtherChannel, and it cannot be a VLAN. • (Optional) [**,** | -] Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. • (Optional) **encapsulation replicate** specifies that the destination interface replicates the source interface encapsulation method. If not selected, the default is to send packets in native form (untagged).
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a VLAN as an RSPAN VLAN

Follow these steps to create a new VLAN, then configure it to be the RSPAN VLAN for the RSPAN session.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan** *vlan-id* **Example:** `Device(config)# vlan 100`	Enters a VLAN ID to create a VLAN, or enters the VLAN ID of an existing VLAN, and enters VLAN configuration mode. The range is 2 to 1001 and 1006 to 4094. The RSPAN VLAN cannot be VLAN 1 (the default VLAN) or VLAN IDs 1002 through 1005 (reserved for Token Ring and FDDI VLANs).
Step 4	**remote-span** **Example:** `Device(config-vlan)# remote-span`	Configures the VLAN as an RSPAN VLAN.
Step 5	**end** **Example:** `Device(config-vlan)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

You must create the RSPAN VLAN in all devices that will participate in RSPAN. If the RSPAN VLAN-ID is in the normal range (lower than 1005) and VTP is enabled in the network, you can create the RSPAN VLAN in one device, and VTP propagates it to the other devices in the VTP domain. For extended-range VLANs (greater than 1005), you must configure RSPAN VLAN on both source and destination devices and any intermediate devices.

Use VTP pruning to get an efficient flow of RSPAN traffic, or manually delete the RSPAN VLAN from all trunks that do not need to carry the RSPAN traffic.

To remove the remote SPAN characteristic from a VLAN and convert it back to a normal VLAN, use the **no remote-span** VLAN configuration command.

To remove a source port or VLAN from the SPAN session, use the **no monitor session** *session_number* **source** {**interface** *interface-id* | **vlan** *vlan-id*} global configuration command. To remove the RSPAN VLAN from the session, use the **no monitor session** *session_number* **destination remote vlan** *vlan-id*.

Creating an RSPAN Source Session

Follow these steps to create and start an RSPAN source session and to specify the monitored source and the destination RSPAN VLAN.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session 1`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source** {**interface** *interface-id* | **vlan** *vlan-id*} [**,** | **-**] [**both** | **rx** | **tx**] **Example:**	Specifies the RSPAN session and the source port (monitored port). • For *session_number*, the range is 1 to 66.

	Command or Action	Purpose
	`Device(config)# monitor session 1 source interface gigabitethernet 1/1 tx`	• Enter a source port or source VLAN for the RSPAN session: • For *interface-id*, specifies the source port to monitor. Valid interfaces include physical interfaces and port-channel logical interfaces (**port-channel** *port-channel-number*). Valid port-channel numbers are 1 to 48. • For *vlan-id*, specifies the source VLAN to monitor. The range is 1 to 4094 (excluding the RSPAN VLAN). A single session can include multiple sources (ports or VLANs), defined in a series of commands, but you cannot combine source ports and source VLANs in one session. • (Optional) [**,** | -]—Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. • (Optional) **both** | **rx** | **tx**—Specifies the direction of traffic to monitor. If you do not specify a traffic direction, the source interface sends both sent and received traffic. • **both**—Monitors both received and sent traffic. • **rx**—Monitors received traffic. • **tx**—Monitors sent traffic.
Step 5	**monitor session** *session_number* **destination remote vlan** *vlan-id* **Example:** `Device(config)# monitor session 1 destination remote vlan 100`	Specifies the RSPAN session, the destination RSPAN VLAN, and the destination-port group. • For *session_number*, enter the number defined in Step 4. • For *vlan-id*, specify the source RSPAN VLAN to monitor.
Step 6	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Specifying VLANs to Filter

Follow these steps to configure the RSPAN source session to limit RSPAN source traffic to specific VLANs.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session 2`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source interface** *interface-id*	Specifies the characteristics of the source port (monitored port) and SPAN session.

	Command or Action	Purpose
	Example: `Device(config)# monitor session 2 source interface gigabitethernet 1/1 rx`	• For *session_number*, the range is 1 to 66. • For *interface-id*, specify the source port to monitor. The interface specified must already be configured as a trunk port.
Step 5	**monitor session** *session_number* **filter vlan** *vlan-id* [**,** | **-**] **Example:** `Device(config)# monitor session 2 filter vlan 1 - 5 , 9`	Limits the SPAN source traffic to specific VLANs. • For *session_number*, enter the session number specified in step 4. • For *vlan-id*, the range is 1 to 4094. • (Optional) **,** | **-** Use a comma (**,**) to specify a series of VLANs or use a hyphen (**-**) to specify a range of VLANs. Enter a space before and after the comma; enter a space before and after the hyphen.
Step 6	**monitor session** *session_number* **destination remote vlan** *vlan-id* **Example:** `Device(config)# monitor session 2 destination remote vlan 902`	Specifies the RSPAN session and the destination remote VLAN (RSPAN VLAN). • For *session_number*, enter the session number specified in Step 4. • For *vlan-id*, specify the RSPAN VLAN to carry the monitored traffic to the destination port.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Creating an RSPAN Destination Session

You configure an RSPAN destination session on a different device; that is, not the device on which the source session was configured.

Follow these steps to define the RSPAN VLAN on that device, to create an RSPAN destination session, and to specify the source RSPAN VLAN and the destination port.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan** *vlan-id* **Example:** `Device(config)# vlan 901`	Specifies the VLAN ID of the RSPAN VLAN created from the source device, and enters VLAN configuration mode. If both devices are participating in VTP and the RSPAN VLAN ID is from 2 to 1005, Steps 3 through 5 are not required because the RSPAN VLAN ID is propagated through the VTP network.
Step 4	**remote-span** **Example:** `Device(config-vlan)# remote-span`	Identifies the VLAN as the RSPAN VLAN.
Step 5	**exit** **Example:** `Device(config-vlan)# exit`	Returns to global configuration mode.
Step 6	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session 1`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.

	Command or Action	Purpose
Step 7	**monitor session** *session_number* **source remote vlan** *vlan-id* **Example:** `Device(config)# monitor session 1 source remote vlan 901`	Specifies the RSPAN session and the source RSPAN VLAN. • For *session_number*, the range is 1 to 66. • For *vlan-id*, specify the source RSPAN VLAN to monitor.
Step 8	**monitor session** *session_number* **destination interface** *interface-id* **Example:** `Device(config)# monitor session 1 destination interface gigabitethernet2/0/1`	Specifies the RSPAN session and the destination interface. • For *session_number*, enter the number defined in Step 7. In an RSPAN destination session, you must use the same session number for the source RSPAN VLAN and the destination port. • For *interface-id*, specify the destination interface. The destination interface must be a physical interface. • Though visible in the command-line help string, **encapsulation replicate** is not supported for RSPAN. The original VLAN ID is overwritten by the RSPAN VLAN ID, and all packets appear on the destination port as untagged.
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 10	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Creating an RSPAN Destination Session and Configuring Incoming Traffic

Follow these steps to create an RSPAN destination session, to specify the source RSPAN VLAN and the destination port, and to enable incoming traffic on the destination port for a network security device (such as a Cisco IDS Sensor Appliance).

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session 2`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source remote vlan** *vlan-id* **Example:** `Device(config)# monitor session 2 source remote vlan 901`	Specifies the RSPAN session and the source RSPAN VLAN. • For *session_number*, the range is 1 to 66. • For *vlan-id*, specify the source RSPAN VLAN to monitor.
Step 5	**monitor session** *session_number* **destination** {**interface** *interface-id* [**,** | -] [**ingress** {**dot1q vlan** *vlan-id* | **untagged vlan** *vlan-id* | **vlan** *vlan-id*}]} **Example:** `Device(config)# monitor session 2 destination interface gigabitethernet 1/1 ingress vlan 6`	Specifies the SPAN session, the destination port, the packet encapsulation, and the incoming VLAN and encapsulation. • For *session_number*, enter the number defined in Step 5. In an RSPAN destination session, you must use the same session number for the source RSPAN VLAN and the destination port.

	Command or Action	Purpose
		• For *interface-id*, specify the destination interface. The destination interface must be a physical interface. • Though visible in the command-line help string, **encapsulation replicate** is not supported for RSPAN. The original VLAN ID is overwritten by the RSPAN VLAN ID, and all packets appear on the destination port as untagged. • (Optional) [**,** | -] Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. • Enter **ingress** with additional keywords to enable forwarding of incoming traffic on the destination port and to specify the encapsulation type: • **dot1q vlan** *vlan-id*—Forwards incoming packets with IEEE 802.1Q encapsulation with the specified VLAN as the default VLAN. • **untagged vlan** *vlan-id* or **vlan** *vlan-id*—Forwards incoming packets with untagged encapsulation type with the specified VLAN as the default VLAN.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring an FSPAN Session

Follow these steps to create a SPAN session, specify the source (monitored) ports or VLANs and the destination (monitoring) ports, and configure FSPAN for the session.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session 2`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source** {**interface** *interface-id* | **vlan** *vlan-id*} [**,** | **-**] [**both** | **rx** | **tx**] **Example:** `Device(config)# monitor session 2 source interface gigabitethernet1/1`	Specifies the SPAN session and the source port (monitored port). • For *session_number*, the range is 1 to 66. • For *interface-id*, specifies the source port to monitor. Valid interfaces include physical interfaces and port-channel logical interfaces (**port-channel** *port-channel-number*). Valid port-channel numbers are 1 to 48. • For *vlan-id*, specify the source VLAN to monitor. The range is 1 to 4094 (excluding the RSPAN VLAN). **Note** A single session can include multiple sources (ports or VLANs) defined in a series of commands, but you cannot combine source ports and source VLANs in one session.

	Command or Action	Purpose
		• (Optional) [**,** | -]—Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. • (Optional) [**both** | **rx** | **tx**]—Specifies the direction of traffic to monitor. If you do not specify a traffic direction, the SPAN monitors both sent and received traffic. • **both**—Monitors both sent and received traffic. This is the default. • **rx**—Monitors received traffic. • **tx**—Monitors sent traffic. **Note** You can use the **monitor session** *session_number* **source** command multiple times to configure multiple source ports.
Step 5	**monitor session** *session_number* **destination** {**interface** *interface-id* [**,** | -] [**encapsulation replicate**]} **Example:** `Device(config)# monitor session 2 destination interface gigabitethernet1/2 encapsulation replicate`	Specifies the SPAN session and the destination port (monitoring port). • For *session_number*, specify the session number entered in Step 4. • For **destination**, specify the following parameters: • For *interface-id*, specify the destination port. The destination interface must be a physical port; it cannot be an EtherChannel, and it cannot be a VLAN. • (Optional) [**,** | -] Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen. • (Optional) **encapsulation replicate** specifies that the destination interface replicates the source interface encapsulation method. If not selected, the default is to send packets in native form (untagged). **Note**

	Command or Action	Purpose
		For local SPAN, you must use the same session number for the source and destination interfaces. You can use **monitor session** *session_number* **destination** command multiple times to configure multiple destination ports.
Step 6	**monitor session** *session_number* **filter** {**ip** | **ipv6** | **mac**} **access-group** {*access-list-number* | *name*} **Example:** `Device(config)# monitor session 2 filter ipv6 access-group 4`	Specifies the SPAN session, the types of packets to filter, and the ACLs to use in an FSPAN session. • For *session_number*, specify the session number entered in Step 4. • For *access-list-number*, specify the ACL number that you want to use to filter traffic. • For *name*, specify the ACL name that you want to use to filter traffic.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 8	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring an FRSPAN Session

Follow these steps to start an RSPAN source session, specify the monitored source and the destination RSPAN VLAN, and configure FRSPAN for the session.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no monitor session** {*session_number* | **all** | **local** | **remote**} **Example:** `Device(config)# no monitor session 2`	Removes any existing SPAN configuration for the session. • For *session_number*, the range is 1 to 66. • **all**—Removes all SPAN sessions. • **local**—Removes all local sessions. • **remote**—Removes all remote SPAN sessions.
Step 4	**monitor session** *session_number* **source** {**interface** *interface-id* | **vlan** *vlan-id*} [**,** | **-**] [**both** | **rx** | **tx**] **Example:** `Device(config)# monitor session 2 source interface gigabitethernet 1/1`	Specifies the SPAN session and the source port (monitored port). • For *session_number*, the range is 1 to 66. • For *interface-id*, specifies the source port to monitor. Valid interfaces include physical interfaces and port-channel logical interfaces (**port-channel** *port-channel-number*). Valid port-channel numbers are 1 to 48. • For *vlan-id*, specify the source VLAN to monitor. The range is 1 to 4094 (excluding the RSPAN VLAN). **Note** A single session can include multiple sources (ports or VLANs) defined in a series of commands, but you cannot combine source ports and source VLANs in one session. • (Optional) [**,** | **-**]—Specifies a series or range of interfaces. Enter a space before and after the comma; enter a space before and after the hyphen.

	Command or Action	Purpose
		• (Optional) [**both** | **rx** | **tx**]—Specifies the direction of traffic to monitor. If you do not specify a traffic direction, the SPAN monitors both sent and received traffic. • **both**—Monitors both sent and received traffic. This is the default. • **rx**—Monitors received traffic. • **tx**—Monitors sent traffic. **Note** You can use the **monitor session** *session_number* **source** command multiple times to configure multiple source ports.
Step 5	**monitor session** *session_number* **destination remote vlan** *vlan-id* **Example:** `Device(config)# monitor session 2 destination remote vlan 5`	Specifies the RSPAN session and the destination RSPAN VLAN. • For *session_number*, enter the number defined in Step 4. • For *vlan-id*, specify the destination RSPAN VLAN to monitor.
Step 6	**vlan** *vlan-id* **Example:** `Device(config)# vlan 10`	Enters the VLAN configuration mode. For *vlan-id*, specify the source RSPAN VLAN to monitor.
Step 7	**remote-span** **Example:** `Device(config-vlan)# remote-span`	Specifies that the VLAN you specified in Step 5 is part of the RSPAN VLAN.
Step 8	**exit** **Example:** `Device(config-vlan)# exit`	Returns to global configuration mode.
Step 9	**monitor session** *session_number* **filter** {**ip** | **ipv6** | **mac**} **access-group** {*access-list-number* | *name*} **Example:** `Device(config)# monitor session 2 filter ip access-group 7`	Specifies the RSPAN session, the types of packets to filter, and the ACLs to use in an FRSPAN session. • For *session_number*, specify the session number entered in Step 4.

	Command or Action	Purpose
		• For *access-list-number*, specify the ACL number that you want to use to filter traffic. • For *name*, specify the ACL name that you want to use to filter traffic.
Step 10	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 11	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 12	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring SPAN and RSPAN Operations

The following table describes the command used to display SPAN and RSPAN operations configuration and results to monitor operations:

Table 166: Monitoring SPAN and RSPAN Operations

Command	Purpose
show monitor	Displays the current SPAN, RSPAN, FSPAN, or FRSPAN configuration.

Configuration Examples for SPAN and RSPAN

The following sections provide configuration examples for SPAN and RSPAN

Example: Configuring Local SPAN

This example shows how to set up SPAN session 1 for monitoring source port traffic to a destination port. First, any existing SPAN configuration for session 1 is deleted, and then bidirectional traffic is mirrored from source Gigabit Ethernet port 1 to destination Gigabit Ethernet port 2, retaining the encapsulation method.

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 1
Device(config)# monitor session 1 source interface gigabitethernet 1/1
Device(config)# monitor session 1 destination interface gigabitethernet 1/1
encapsulation replicate
Device(config)# end
```

This example shows how to remove port 1 as a SPAN source for SPAN session 1:

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 1 source interface gigabitethernet 1/1
Device(config)# end
```

This example shows how to disable received traffic monitoring on port 1, which was configured for bidirectional monitoring:

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 1 source interface gigabitethernet 1/1 rx
```

The monitoring of traffic received on port 1 is disabled, but traffic sent from this port continues to be monitored.

This example shows how to remove any existing configuration on SPAN session 2, configure SPAN session 2 to monitor received traffic on all ports belonging to VLANs 1 through 3, and send it to destination Gigabit Ethernet port 2. The configuration is then modified to also monitor all traffic on all ports belonging to VLAN 10.

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 2
Device(config)# monitor session 2 source vlan 1 - 3 rx

Device(config)# monitor session 2 destination interface gigabitethernet 1/2
Device(config)# monitor session 2 source vlan 10
Device(config)# end
```

This example shows how to remove any existing configuration on SPAN session 2, configure SPAN session 2 to monitor received traffic on Gigabit Ethernet source port 1, and send it to destination Gigabit Ethernet port 2 with the same egress encapsulation type as the source port, and to enable ingress forwarding with VLAN 6 as the default ingress VLAN:

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 2
Device(config)# monitor session 2 source gigabitethernet0/1 rx
Device(config)# monitor session 2 destination interface gigabitethernet 1/1 encapsulation
replicate ingress vlan 6
Device(config)# end
```

This example shows how to remove any existing configuration on SPAN session 2, configure SPAN session 2 to monitor traffic received on Gigabit Ethernet trunk port 2, and send traffic for only VLANs 1 through 5 and VLAN 9 to destination Gigabit Ethernet port 1:

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 2
Device(config)# monitor session 2 source interface gigabitethernet 1/1 rx
Device(config)# monitor session 2 filter vlan 1 - 5 , 9
Device(config)# monitor session 2 destination interface gigabitethernet 1/1
Device(config)# end
```

Examples: Creating an RSPAN VLAN

This example shows how to create the RSPAN VLAN 901:

```
Device> enable
Device# configure terminal
Device(config)# vlan 901
Device(config-vlan)# remote span
Device(config-vlan)# end
```

This example shows how to remove any existing RSPAN configuration for session 1, configure RSPAN session 1 to monitor multiple source interfaces, and configure the destination as RSPAN VLAN 901:

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 1
Device(config)# monitor session 1 source interface gigabitethernet 1/1 tx
Device(config)# monitor session 1 source interface gigabitethernet 1/1 rx
Device(config)# monitor session 1 source interface port-channel 2
Device(config)# monitor session 1 destination remote vlan 901
Device(config)# end
```

This example shows how to remove any existing configuration on RSPAN session 2, configure RSPAN session 2 to monitor traffic received on trunk port 2, and send traffic for only VLANs 1 through 5 and 9 to destination RSPAN VLAN 902:

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 2
Device(config)# monitor session 2 source interface gigabitethernet 1/2 rx
Device(config)# monitor session 2 filter vlan 1 - 5 , 9
Device(config)# monitor session 2 destination remote vlan 902
Device(config)# end
```

This example shows how to configure VLAN 901 as the source remote VLAN and port 1 as the destination interface:

```
Device> enable
Device# configure terminal
Device(config)# monitor session 1 source remote vlan 901
Device(config)# monitor session 1 destination interface gigabitethernet 1/1
Device(config)# end
```

This example shows how to configure VLAN 901 as the source remote VLAN in RSPAN session 2, to configure Gigabit Ethernet source port 2 as the destination interface, and to enable forwarding of incoming traffic on the interface with VLAN 6 as the default receiving VLAN:

```
Device> enable
Device# configure terminal
Device(config)# monitor session 2 source remote vlan 901
Device(config)# monitor session 2 destination interface gigabitethernet 1/1 ingress vlan 6
Device(config)# end
```

Here are the list of debug CLIs to debug the monitor:

```
Device(config)# debug monitor
  all             All SPAN debugging messages
  capture         Show Capture tracing
  errors          Show SPAN error detail
  erspan          Show ERSPAN tracing
  idb-update      Show SPAN IDB update traces
  info            Show SPAN Informational tracing
  list            Show SPAN port and VLAN list tracing
  notifications   Show SPAN notifications
  platform        Show SPAN platform tracing
  redundancy      Show SPAN Redundancy tracing
  requests        Show SPAN requests
  scp             Show SPAN SCP tracing
  snmp            Show SPAN SNMP tracing
```

CHAPTER 164

ERSPAN

ERSPAN

The Cisco Encapsulated Remote Switched Port Analyzer (ERSPAN) feature allows you to monitor traffic on ports or VLANs, and send the monitored traffic to destination ports over a Layer 3 (IP) network using Generic Routing Encapsulation (GRE) encapsulation. ERSPAN sends traffic to a network analyzer, such as a Switch Probe device or a Remote Monitoring (RMON) probe. ERSPAN supports source ports, source VLANs, and destination ports on different devices, which help remote monitoring of multiple devices across a network.

ERSPAN supports encapsulated packets of up to 9180 bytes. ERSPAN consists of an ERSPAN source session, routable ERSPAN GRE-encapsulated traffic, and an ERSPAN destination session.

You can configure an ERSPAN source session, an ERSPAN destination session, or both on a device. A device on which only an ERSPAN source session is configured is called an ERSPAN source device. A device on which only an ERSPAN destination session is configured is called an ERSPAN termination device. A device can act as both; an ERSPAN source device and a termination device.

Over-subscription of traffic can lead to a drop in management traffic on the destination device. To avoid over-subscription, ensure that the destination session is configured and is working on the destination device, before configuring a source session on the source device.

For a source port or a source VLAN, the ERSPAN can monitor the ingress, egress, or both ingress and egress traffic. By default, ERSPAN monitors all traffic, including multicast, and Bridge Protocol Data Unit (BPDU) frames.

A device supports up to 66 sessions. A maximum of eight source sessions can be configured and the remaining sessions can be configured as RSPAN destinations sessions. A source session can be a local SPAN source session or an RSPAN source session or an ERSPAN source session.

An ERSPAN source session is defined by the following parameters:

- A session ID.
- ERSPAN flow ID.
- List of source ports or source VLANs that are monitored by the session.

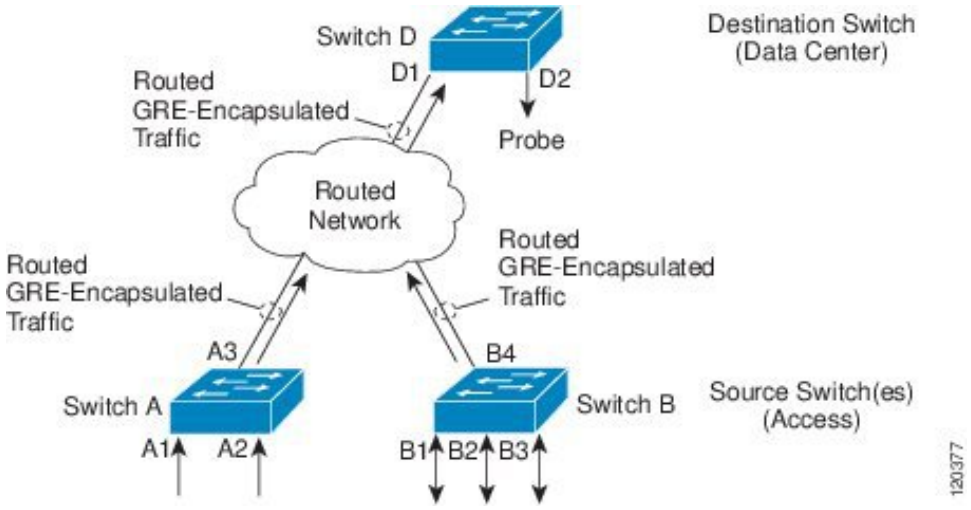
- Optional attributes, such as, IP type of service (ToS) and IP Time to Live (TTL), related to the Generic Routing Encapsulation (GRE) envelope.
- The destination and origin IP addresses. These are used as the destination and source IP addresses of the GRE envelope for the captured traffic, respectively.

Note

- ERSPAN source sessions do not copy ERSPAN GRE-encapsulated traffic from source ports. Each ERSPAN source session can have either ports or VLANs as sources, but not both.
- IPv4 delivery and transport headers are supported; including Type-II and Type-III headers.

  Port channel and switch virtual interface (SVI) are supported.

Figure 151: ERSPAN Configuration



Information About Configuring ERSPAN

The following sections provide information about configuring ERSPAN.

Restrictions for Configuring ERSPAN

The following restrictions apply for this feature:

- Truncation is supported only on IPv4 spanned packets and not on Layer 2 packets without an IP header.
- An ERSPAN destination interface can be part of only one session. The same destination interface cannot be configured for multiple ERSPANs/SPANs.
- You can configure either a list of ports or a list of VLANs as a source, but cannot configure both for a given session.
- Filter IP/MAC/VLAN access-group and filter SGT cannot be configured at the same time.

- When a session is configured through the ERSPAN CLI, the session ID and the session type cannot be changed. To change them, you must use the **no** form of the commands to remove the session and then reconfigure it.
- ERSPAN source sessions do not copy locally-sourced RSPAN VLAN traffic from source trunk ports that carry RSPAN VLANs.
- ERSPAN source sessions do not copy locally-sourced ERSPAN Generic routing encapsulation (GRE)-encapsulated traffic from source ports.
- Disabling the **ip routing** command for IPv4 connections stops ERSPAN traffic flow to the destination port.

ERSPAN Sources

The Cisco ERSPAN feature supports the following sources:

- Source ports: A source port that is monitored for traffic analysis. Source ports in any VLAN can be configured and trunk ports can be configured as source ports along with nontrunk source ports.
- Source VLANs: A VLAN that is monitored for traffic analysis.

ERSPAN Destination Ports

A destination port is a Layer 2 or Layer 3 port to which ERSPAN source sends traffic for analysis.

When you configure a port as a destination port, it can no longer receive any traffic. The port is dedicated for use only by the ERSPAN feature. An ERSPAN destination port does not forward any traffic except that required for the ERSPAN session. You can configure trunk ports as destination ports, which allows destination trunk ports to transmit encapsulated traffic.

SGT-Based ERSPAN

A Security Group Tag (SGT) is a 16-bit value that the Cisco Identity Services Engine (ISE) assigns to the user or endpoint session upon login. The network infrastructure views the SGT as another attribute to assign to the session and inserts the Layer 2 tag to all traffic from that session. A platform can support a maximum of 50 SGT policies per session.

On an existing flow-based SPAN (FSPAN) or VLAN filter session, SGT filtering configurations are not allowed.

Prerequisites for Configuring ERSPAN

Apply the Access control list (ACL) filter before sending the monitored traffic on to the tunnel.

How to Configure ERSPAN

The following sections provide information about how to configure ERSPAN.

Configuring an ERSPAN Source Session

The ERSPAN source session defines the session configuration parameters and the ports or VLANs to be monitored. To define an IPv4 ERSPAN source session, complete the following procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**monitor session** *span-session-number* **type erspan-source** **Example:** `Device(config)# monitor session 1 type erspan-source`	Defines an ERSPAN source session using the session ID and the session type, and enters ERSPAN monitor source session configuration mode. • The *span-session-number* argument range is from 1 to 66. The same session number cannot be used more than once. • The session IDs for source sessions or destination sessions are in the same global ID space, so each session ID is globally unique for both session types. • The session ID (configured by the *span-session-number* argument) and the session type (configured by the **erspan-source** keyword) cannot be changed once entered. Use the **no** form of this command to remove the session and then re-create the session, with a new session ID or a new session type.
Step 4	**description** *string* **Example:** `Device(config-mon-erspan-src)# description source1`	(Optional) Describes the ERSPAN source session. • The *string* argument can be up to 240 characters and cannot contain special characters or spaces.
Step 5	[**no**] **header-type 3** **Example:** `Device(config-mon-erspan-src)# header-type 3`	(Optional) Configures a switch to Type-III ERSPAN header. The default type is Type-II ERSPAN header.

	Command or Action	Purpose
Step 6	**source** {**interface** *interface-type interface-number* | **vlan** *vlan-id*} [**,** | **-** | **both** | **rx** | **tx**] **Example:** `Device(config-mon-erspan-src)# source interface gigabitethernet 1/1 rx`	Configures the source interface or the VLAN, and the traffic direction to be monitored.
Step 7	**filter** {**ip access-group** {*standard-access-list* | *expanded-access-list* | *acl-name* } | **mac access-group** *acl-name* | **sgt** *sgt-ID* [**,** | **-**] | **vlan** *vlan-ID* [**,** | **-**]} **Example:** `Switch(config-mon-erspan-src)# filter vlan 3`	(Optional) Configures source VLAN filtering when the ERSPAN source is a trunk port. **Note** You cannot include source VLANs and filter VLANs in the same session.
Step 8	**destination** **Example:** `Device(config-mon-erspan-src)# destination`	Enters ERSPAN source session destination configuration mode.
Step 9	**erspan-id** *erspan-flow-id* **Example:** `Device(config-mon-erspan-src-dst)# erspan-id 100`	Configures the ID used by source and destination sessions to identify the ERSPAN traffic, which must also be entered in the ERSPAN destination session configuration.
Step 10	**ip address** *ip-address* **Example:** `Device(config-mon-erspan-src-dst)# ip address 10.1.0.2`	Configures the IP address that is used as the destination of the ERSPAN traffic.
Step 11	**ip dscp** *dscp-value* **Example:** `Device(config-mon-erspan-src-dst)# ip dscp 10`	(Optional) Enables the use of IP differentiated services code point (DSCP) for packets that originate from a circuit emulation (CEM) channel.
Step 12	**ip ttl** *ttl-value* **Example:** `Device(config-mon-erspan-src-dst)# ip ttl 32`	(Optional) Configures the IP TTL value of packets in the ERSPAN traffic.
Step 13	**mtu** *mtu-size* **Example:** `Device(config-mon-erspan-src-dst)# mtu 512`	Configures the MTU size for truncation. Any ERSPAN packet that is larger than the configured MTU size is truncated to the configured size. The MTU size range is 176 to 9000 bytes. The default value is 9000 bytes.

	Command or Action	Purpose
Step 14	**origin ip-address** *ip-address* **Example:** `Device(config-mon-erspan-src-dst)# origin ip address 10.10.0.1`	Configures the IP address used as the source of the ERSPAN traffic.
Step 15	**vrf** *vrf-id* **Example:** `Device(config-mon-erspan-src-dst)# vrf 1`	(Optional) Configures the VRF name to use instead of the global routing table.
Step 16	**exit** **Example:** `Device(config-mon-erspan-src-dst)# exit`	Exits ERSPAN source session destination configuration mode, and returns to ERSPAN source session configuration mode.
Step 17	**no shutdown** **Example:** `Device(config-mon-erspan-src)# no shutdown`	Enables the configured sessions on an interface.
Step 18	**end** **Example:** `Device(config-mon-erspan-src)# end`	Exits ERSPAN source session configuration mode, and returns to privileged EXEC mode.

Configuring an ERSPAN Destination Session

The ERSPAN destination session defines the session configuration parameters and the ports that receive the monitored traffic. To define an IPv4 ERSPAN destination session, complete the following procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**monitor session** *session-number* **type erspan-destination** **Example:** `Device(config)# monitor session 1 type erspan-destination`	Defines an ERSPAN destination session using the session ID and the session type, and enters ERSPAN monitor destination session configuration mode.

	Command or Action	Purpose
		• The *session-number* argument range is from 1 – 66. The session number must be unique and cannot be used more than once. • The session IDs for source sessions or destination sessions are in the same global ID space, so each session ID is globally unique for both session types. • The session ID (configured by the *session-number* argument) and the session type (configured by the **erspan-destination**) cannot be changed once entered. Use the **no** form of this command to remove the session, and then recreate the session with a new session ID or a new session type.
Step 4	**description** *string* **Example:** `Device(config-mon-erspan-dst)# description source1`	(Optional) Describes the ERSPAN destination session. • The *string* argument can be up to 240 characters in length and cannot contain special characters or spaces.
Step 5	**destination interface** *interface-type* **Example:** `Device(config-mon-erspan-dst)# destination interface GigabitEthernet1/1`	Associates the ERSPAN destination session number with source ports, and selects the traffic direction to be monitored.
Step 6	**source** **Example:** `Device(config-mon-erspan-dst)# source`	Enters ERSPAN destination session source configuration mode.
Step 7	**erspan-id** *erspan-flow-id* **Example:** `Device(config-mon-erspan-dst-src)# erspan-id 100`	Configures the ID used by source and destination sessions to identify the ERSPAN traffic, which must also be entered in the ERSPAN source session configuration.
Step 8	**ip address** *ip-address* [**force**] **Example:** `Device(config-mon-erspan-dst-src)# ip address 10.1.0.2`	Configures the IP address that is used as the destination of the ERSPAN traffic. • This IP address must be an address on a local interface or loopback interface, and match the address on the destination switch. • The **ip address** *ip-address* **force** command changes the destination IP

	Command or Action	Purpose
		address for all ERSPAN destination sessions.
Step 9	**vrf** *vrf-id* **Example:** `Device(config-mon-erspan-dst-src)# vrf 1`	(Optional) Configures the VRF name to use instead of the global routing table.
Step 10	**no shutdown** **Example:** `Device(config-mon-erspan-dst-src)# no shutdown`	Enables the configured sessions on an interface.
Step 11	**end** **Example:** `Device(config-mon-erspan-dst-src)# end`	Exits ERSPAN destination session source configuration mode, and returns to privileged EXEC mode.

Configuration Examples for ERSPAN

The following sections provide configuration examples for ERSPAN.

Example: Configuring an ERSPAN Source Session

The following example shows how to configure an ERSPAN source session:

```
Device> enable
Device# configure terminal
Device(config)# monitor session 1 type erspan-source
Device(config-mon-erspan-src)# description source1
Device(config-mon-erspan-src)# source interface GigabitEthernet 1/1 rx
Device(config-mon-erspan-src)# source interface GigabitEthernet 1/1 - 8 tx
Device(config-mon-erspan-src)# source interface GigabitEthernet 1/1
Device(config-mon-erspan-src)# destination
Device(config-mon-erspan-src-dst)# erspan-id 100
Device(config-mon-erspan-src-dst)# ip address 10.1.0.2
Device(config-mon-erspan-src-dst)# ip dscp 10
Device(config-mon-erspan-src-dst)# ip ttl 32
Device(config-mon-erspan-src-dst)# mtu 512
Device(config-mon-erspan-src-dst)# origin ip address 10.10.0.1
Device(config-mon-erspan-src-dst)# vrf monitoring
Device(config-mon-erspan-src-dst)# exit
Device(config-mon-erspan-src)# no shutdown
Device(config-mon-erspan-src)# end
```

Example: Configuring an ERSPAN Destination Session

The following example shows how to configure an ERSPAN destination session:

```
Device(config)# monitor session 2 type erspan-destination
Device(config-mon-erspan-dst)# destination interface GigabitEthernet1/1
Device(config-mon-erspan-dst)# destination interface GigabitEthernet1/1
```

```
Device(config-mon-erspan-dst)# source
Device(config-mon-erspan-dst-src)# erspan-id 100
Device(config-mon-erspan-dst-src)# ip address 10.1.0.2
```

The following example shows how to configure a source VRF for an ERSPAN destination session:

```
Device(config)# monitor session 2 type erspan-destination
Device(config-mon-erspan-dst)# destination interface GigabitEthernet1/1
Device(config-mon-erspan-dst)# destination interface GigabitEthernet1/1
Device(config-mon-erspan-dst)# source
Device(config-mon-erspan-dst-src)# erspan-id 100
Device(config-mon-erspan-dst-src)# ip address 10.1.0.2
Device(config-mon-erspan-dst-src)# vrf 1
```

Verifying ERSPAN

To verify the ERSPAN configuration, use the following commands:

The following is sample output from the **show monitor session erspan-source** command:

```
Device# show monitor session erspan-source

Type : ERSPAN Source Session
Status : Admin Enabled
Source Ports :
RX Only : Gi1/1
Destination IP Address : 192.0.2.1
Destination ERSPAN ID : 110
Origin IP Address : 10.10.10.216
 Flow Label : None


Device# show monitor session 53

Session 53
----------
Type                    : ERSPAN Source Session
Status                  : Admin Enabled
Source Ports            :
MTU                     : 9000
```

The following is sample output from the **show platform software monitor session** command:

```
Device# show platform software monitor session 53

Span Session 53 (FED Session 0):
Type: ERSPAN Source
Prev type: Unknown
Ingress Src Ports:
Egress Src Ports:
Ingress Local Src Ports: (null)
Egress Local Src Ports: (null)
Destination Ports:
Ingress Src Vlans:
Egress Src Vlans:
Ingress Up Src Vlans: (null)
Egress Up Src Vlans: (null)
Src Trunk filter Vlans:
RSPAN dst vlan: 0
```

```
RSPAN src vlan: 0
RSPAN src vlan sav: 0
Dest port encap = 0x0000
Dest port ingress encap = 0x0000
Dest port ingress vlan = 0x0
SrcSess: 1 DstSess: 0 DstPortCfgd: 0 RspnDstCfg: 0 RspnSrcVld: 0
DstCliCfg: 0 DstPrtInit: 0 PsLclCfgd: 0
Flags: 0x00000000
Remote dest port: 0 Dest port group: 0
FSPAN disabled
FSPAN not notified
ERSPAN Id : 0
ERSPAN Org Ip: 0.0.0.0
ERSPAN Dst Ip: 0.0.0.0
ERSPAN Ip Ttl: 255
ERSPAN DSCP : 0
ERSPAN MTU : 1500 >>>>
ERSPAN VRFID : 0
ERSPAN State : Disabled
ERSPAN Tun id: 61
ERSPAN header-type: 2
ERSPAN SGT :
```

The following is sample output from the **show monitor session erspan-source detail** command:

```
Device# show monitor session erspan-source detail

Type                    : ERSPAN Source Session
Status                  : Admin Enabled
Description             : -
Source Ports            :
    RX Only             : None
    TX Only             : None
    Both                : None
Source Subinterfaces    :
    RX Only             : None
    TX Only             : None
    Both                : None
Source VLANs            :
    RX Only             : None
    TX Only             : None
    Both                : None
Source Drop-cause       : None
Source EFPs             :
    RX Only             : None
    TX Only             : None
    Both                : None
Source RSPAN VLAN       : None
Destination Ports       : None
Filter VLANs            : None
Filter SGT              : None
Dest RSPAN VLAN         : None
IP Access-group         : None
MAC Access-group        : None
IPv6 Access-group       : None
Filter access-group  :None
smac for wan interface   : None
dmac for wan interface   : None
Destination IP Address   : 192.0.2.1
Destination IPv6 Address : None
Destination IP VRF       : None
MTU                      : 1500
Destination ERSPAN ID    : 251
Origin IP Address        : 10.10.10.216
```

```
Origin IPv6 Address      : None
IP QOS PREC              : 0
IPv6 Flow Label          : None
IP TTL                   : 255
ERSPAN header-type       : 3
```

The following output from the **show capability feature monitor erspan-source** command displays information about the configured ERSPAN source sessions:

```
Device# show capability feature monitor erspan-source

ERSPAN Source Session:ERSPAN Source Session Supported: TRUE
No of Rx ERSPAN source session: 8
No of Tx ERSPAN source session: 8
ERSPAN Header Type supported: II and III
ACL filter Supported: TRUE
SGT filter Supported: TRUE
Fragmentation Supported: TRUE
Truncation Supported: FALSE
Sequence number Supported: FALSE
QOS Supported: TRUE
```

The following output from the **show capability feature monitor erspan-destination** command displays all the configured global built-in templates:

```
Device# show capability feature monitor erspan-destination

ERSPAN Destination Session:ERSPAN Destination Session Supported: TRUE
Maximum No of ERSPAN destination session: 8
ERSPAN Header Type supported: II and III
```

CHAPTER 165

Configuring Packet Capture

Prerequisites for Configuring Packet Capture

Packet capture is supported.

The following section provides information about the prerequisites for configuring packet capture.

Prerequisites for Configuring Embedded Packet Capture

The Embedded Packet Capture (EPC) software subsystem consumes CPU and memory resources during its operation. You must have adequate system resources for different types of operations. The following table provides some guidelines for using the system resources.

Table 167: System Requirements for the EPC Subsystem

System Resources	Requirements
Hardware	CPU utilization requirements are platform-dependent.
Memory	The DRAM stores the packet buffer. The size of the packet buffer is user specified.
Disk space	Packets can be exported to external devices. No intermediate storage on flash disk is required.

Restrictions for Configuring Embedded Packet Capture

- Layer 2 EtherChannels aren't supported.
- You can't use VRFs, management ports, and private VLANs as attachment points.

- Embedded Packet Capture (EPC) isn't supported on logical ports, which includes port channels and switch virtual interfaces (SVIs). It's supported only on physical ports.
- A VLAN interface that is in shutdown state doesn't support EPC.
- If you change interface from switch port to routed port (Layer 2 to Layer 3) or the opposite way, you must delete the capture point and create a new one, once the interface comes back up. Stop/start the capture point won't work.
- Packets captured in the output direction of an interface might not reflect the changes made by the device rewrite. This includes TTL, VLAN tag, CoS, checksum, MAC addresses, DSCP, precedent, UP, etc.
- Even though the minimum configurable duration for packet capture is 1 second, packet capture works for a minimum of 2 seconds.
- It's not possible to modify a capture point parameter when a capture is already active or has started.
- EPC captures multicast packets only on ingress and doesn't capture the replicated packets on egress.
- The Rewrite information of both ingress and egress packets aren't captured.
- CPU-injected packets are considered control plane packets. Therefore, these types of packets won't be captured on an interface egress capture.
- Control plane packets aren't rate limited and performance impacting. Use filters to limit control plane packet capture.
- DNA Advantage supports decoding of protocols such as Control and Provisioning of Wireless Access Points (CAPWAP).
- You can define up to eight capture points, but only one can be active at a time. Stop one before you can start the other.
- MAC filter won't capture IP packets even if it matches the MAC address. This applies to all interfaces (Layer 2 switch port, Layer 3 routed port).
- MAC ACL is only used for non-IP packets such as ARP. It won't be supported on a Layer 3 port or SVI.
- MAC filter can't capture Layer 2 packets (ARP) on Layer 3 interfaces.
- VACL doesn't support IPv6-based ACLs.
- EPC cannot capture based on the underlying routing protocols in MPLS packets.

Information About Packet Capture

The Packet Capture feature is an onboard packet capture facility that allows network administrators to capture packets flowing to, through, and from the device. You can analyze them locally or save and export them for offline analysis by using Embedded Packet Capture (EPC). This feature simplifies network operations by allowing devices to become active participants in the management and operation of the network. This feature facilitates troubleshooting by gathering information about the packet format. This feature also facilitates application analysis and security.

About Embedded Packet Capture

EPC provides an embedded systems management facility that helps in tracing and troubleshooting packets. This feature allows network administrators to capture data packets flowing through, to, and from a Cisco device. The network administrator may define the capture buffer size and type (circular, or linear) and the maximum number of bytes of each packet to capture. You can throttle the packet capture rate using further administrative controls. For example, You can filter the packets using an Access Control List. You can further define the controls by specifying a maximum packet capture rate or by specifying a sampling interval.

EPC is supported on an interface in shutdown state. This is useful in capturing packets on an interface as it's being brought up.

Benefits of Embedded Packet Capture

- Ability to capture IPv4 and IPv6 packets in the device, and also capture non-IP packets with MAC filter or match any MAC address.
- Extensible infrastructure for enabling packet capture points. A capture point is a traffic transit point where a packet is captured and associated with a buffer.
- Facility to export the packet capture in packet capture file (PCAP) format suitable for analysis using any external tool.
- Methods to decode data packets captured with varying degrees of detail.

Packet Data Capture

Packet data capture is the capture of data packets that are then stored in a buffer. You can define packet data captures by providing unique names and parameters.

You can perform the following actions on the capture:

- Activate captures at any interface.
- Apply access control lists (ACLs) or class maps to capture points.

Note Network Based Application Recognition (NBAR) and MAC-style class map is not supported.

- Destroy captures.
- Specify buffer storage parameters such as size and type. The size ranges from 1 MB to 100 MB. The default option for the buffer is linear and the other option for the buffer is circular.
- Specify match criteria that includes information about the protocol, IP address or port address.

How to Implement Embedded Packet Capture

The following sections provide information on how to implement EPC.

Managing Packet Data Capture

To manage Packet Data Capture in the buffer mode, perform the following steps:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**monitor capture** *capture-name* **access-list** *access-list-name* **Example:** `Device# monitor capture mycap access-list v4acl`	Configures a monitor capture specifying an access list as the core filter for the packet capture.
Step 3	**monitor capture** *capture-name* **limit duration** *seconds* **Example:** `Device# monitor capture mycap limit duration 1000`	Configures monitor capture limits.
Step 4	**monitor capture** *capture-name* **interface** *interface-name* **both** **Example:** `Device# monitor capture mycap interface GigabitEthernet 1/1 both`	Configures monitor capture specifying an attachment point and the packet flow direction.
Step 5	**monitor capture** *capture-name* **buffer circular size** *bytes* **Example:** `Device# monitor capture mycap buffer circular size 10`	Configures a buffer to capture packet data.
Step 6	**monitor capture** *capture-name* **start** **Example:** `Device# monitor capture mycap start`	Starts the capture of packet data at a traffic trace point into a buffer.
Step 7	**monitor capture** *capture-name* **stop** **Example:** `Device# monitor capture mycap stop`	Stops the capture of packet data at a traffic trace point.

	Command or Action	Purpose
Step 8	**monitor capture** *capture-name* **export** *file--location/file-name* **Example:** `Device# monitor capture mycap export tftp://10.1.88.9/mycap.pcap`	Exports captured data for analysis.
Step 9	**end** **Example:** `Device# end`	Returns to privileged EXEC mode.

Monitoring and Maintaining Captured Data

Perform this task to monitor and maintain the packet data captured. Capture buffer details and capture point details are displayed.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**show monitor capture** *capture-buffer-name* **buffer dump** **Example:** `Device# show monitor capture mycap buffer dump`	(Optional) Displays a hexadecimal dump of captured packet and its metadata.
Step 3	**show monitor capture** *capture-buffer-name* **parameter** **Example:** `Device# show monitor capture mycap parameter`	(Optional) Displays a list of commands that were used to specify the capture.
Step 4	**debug epc capture-point** **Example:** `Device# debug epc capture-point`	(Optional) Enables packet capture point debugging.

	Command or Action	Purpose
Step 5	**debug epc provision** **Example:** `Device# debug epc provision`	(Optional) Enables packet capture provisioning debugging.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuration Examples for Embedded Packet Capture

Example: Managing Packet Data Capture

The following example shows how to manage packet data capture:

```
Device> enable
Device# monitor capture mycap access-list v4acl
Device# monitor capture mycap limit duration 1000
Device# monitor capture mycap interface GigabitEthernet 1/1 both
Device# monitor capture mycap buffer circular size 10
Device# monitor capture mycap start
Device# monitor capture mycap stop
Device# monitor capture mycap export tftp://10.1.88.9/mycap.pcap
Device# end
```

Example: Monitoring and Maintaining Captured Data

The following example shows how to dump packets in ASCII format:

```
Device# show monitor capture mycap buffer dump
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

0
0000: 01005E00 00020000 0C07AC1D 080045C0 ..^...........E.
0010: 00300000 00000111 CFDC091D 0002E000 .0..............
0020: 000207C1 07C1001C 802A0000 10030AFA .........*......
0030: 1D006369 73636F00 0000091D 0001 ..example.......
1
0000: 01005E00 0002001B 2BF69280 080046C0 ..^.....+.....F.
0010: 00200000 00000102 44170000 0000E000 . ......D.......
0020: 00019404 00001700 E8FF0000 0000 ..............
2
0000: 01005E00 0002001B 2BF68680 080045C0 ..^.....+.....E.
0010: 00300000 00000111 CFDB091D 0003E000 .0..............
0020: 000207C1 07C1001C 88B50000 08030A6E ...............n
0030: 1D006369 73636F00 0000091D 0001 ..example.......
3
0000: 01005E00 000A001C 0F2EDC00 080045C0 ..^...........E.
0010: 003C0000 00000258 CE7F091D 0004E000 .<.....X........
0020: 000A0205 F3000000 00000000 00000000 ................
```

```
0030: 00000000 00D10001 000C0100 01000000 ................
0040: 000F0004 00080501 0300
```

The following example shows how to display the list of commands used to configure the capture named mycap:

```
Device# show monitor capture mycap parameter
monitor capture mycap interface GigabitEthernet 1/1 both
monitor capture mycap match any
monitor capture mycap buffer size 10
monitor capture mycap limit pps 1000
```

The following example shows how to debug the capture point:

```
Device# debug epc capture-point
EPC capture point operations debugging is on

Device# monitor capture mycap start
*Jun 4 14:17:15.463: EPC CP: Starting the capture cap1
*Jun 4 14:17:15.463: EPC CP: (brief=3, detailed=4, dump=5) = 0
*Jun 4 14:17:15.463: EPC CP: final check before activation
*Jun 4 14:17:15.463: EPC CP: setting up c3pl infra
*Jun 4 14:17:15.463: EPC CP: Setup c3pl acl-class-policy
*Jun 4 14:17:15.463: EPC CP: Creating a class
*Jun 4 14:17:15.464: EPC CP: Creating a class : Successful
*Jun 4 14:17:15.464: EPC CP: class-map Created
*Jun 4 14:17:15.464: EPC CP: creating policy-name epc_policy_cap1
*Jun 4 14:17:15.464: EPC CP: Creating Policy epc_policy_cap1 of type 49 and client type 21
*Jun 4 14:17:15.464: EPC CP: Storing a Policy
*Jun 4 14:17:15.464: EPC CP: calling ppm_store_policy with epc_policy
*Jun 4 14:17:15.464: EPC CP: Creating Policy : Successful
*Jun 4 14:17:15.464: EPC CP: policy-map created
*Jun 4 14:17:15.464: EPC CP: creating filter for ANY
*Jun 4 14:17:15.464: EPC CP: Adding acl to class : Successful
*Jun 4 14:17:15.464: EPC CP: Setup c3pl class to policy
*Jun 4 14:17:15.464: EPC CP: Attaching Class to Policy
*Jun 4 14:17:15.464: EPC CP: Attaching epc_class_cap1 to epc_policy_cap1
*Jun 4 14:17:15.464: EPC CP: Attaching Class to Policy : Successful
*Jun 4 14:17:15.464: EPC CP: setting up c3pl qos
*Jun 4 14:17:15.464: EPC CP: DBG> Set packet rate limit to 1000
*Jun 4 14:17:15.464: EPC CP: creating action for policy_map epc_policy_cap1 class_map
epc_class_cap1
*Jun 4 14:17:15.464: EPC CP: DBG> Set packet rate limit to 1000
*Jun 4 14:17:15.464: EPC CP: Activating Interface GigabitEthernet1/1 direction both
*Jun 4 14:17:15.464: EPC CP: Id attached 0
*Jun 4 14:17:15.464: EPC CP: inserting into active lists
*Jun 4 14:17:15.464: EPC CP: Id attached 0
*Jun 4 14:17:15.465: EPC CP: inserting into active lists
*Jun 4 14:17:15.465: EPC CP: Activating Vlan
*Jun 4 14:17:15.465: EPC CP: Deleting all temp interfaces
*Jun 4 14:17:15.465: %BUFCAP-6-ENABLE: Capture Point cap1 enabled.
*Jun 4 14:17:15.465: EPC CP: Active Capture 1

Device# monitor capture mycap1 stop
*Jun 4 14:17:31.963: EPC CP: Stopping the capture cap1
*Jun 4 14:17:31.963: EPC CP: Warning: unable to unbind capture cap1
*Jun 4 14:17:31.963: EPC CP: Deactivating policy-map
*Jun 4 14:17:31.963: EPC CP: Policy epc_policy_cap1
*Jun 4 14:17:31.964: EPC CP: Deactivating policy-map Successful
*Jun 4 14:17:31.964: EPC CP: removing povision feature
*Jun 4 14:17:31.964: EPC CP: Found action for policy-map epc_policy_cap1 class-map
epc_class_cap1
*Jun 4 14:17:31.964: EPC CP: cleanning up c3pl infra
*Jun 4 14:17:31.964: EPC CP: Removing Class epc_class_cap1 from Policy
*Jun 4 14:17:31.964: EPC CP: Removing Class from epc_policy_cap1
```

```
*Jun 4 14:17:31.964: EPC CP: Successfully removed
*Jun 4 14:17:31.964: EPC CP: Removing acl mac from class
*Jun 4 14:17:31.964: EPC CP: Removing acl from class : Successful
*Jun 4 14:17:31.964: EPC CP: Removing all policies
*Jun 4 14:17:31.964: EPC CP: Removing Policy epc_policy_cap1
*Jun 4 14:17:31.964: EPC CP: Removing Policy : Successful
*Jun 4 14:17:31.964: EPC CP: Removing class epc_class_cap1
*Jun 4 14:17:31.965: EPC CP: Removing class : Successful
*Jun 4 14:17:31.965: %BUFCAP-6-DISABLE: Capture Point cap1 disabled.
*Jun 4 14:17:31.965: EPC CP: Active Capture 0
```

The following example shows how to debug the Embedded Packet Capture (EPC) provisioning:

```
Device# debug epc provision
EPC provisionioning debugging is on

Device# monitor capture mycap start
*Jun 4 14:17:54.991: EPC PROV: No action found for policy-map epc_policy_cap1 class-map
epc_class_cap1
*Jun 4 14:17:54.991: EPC PROV:
*Jun 4 14:17:54.991: Attempting to install service policy epc_policy_cap1
*Jun 4 14:17:54.992: EPC PROV: Attached service policy to epc idb subblock
*Jun 4 14:17:54.992: EPC PROV: Successful. Create feature object
*Jun 4 14:17:54.992: EPC PROV:
*Jun 4 14:17:54.992: Attempting to install service policy epc_policy_cap1
*Jun 4 14:17:54.992: EPC PROV: Successful. Create feature object
*Jun 4 14:17:54.992: %BUFCAP-6-ENABLE: Capture Point cap1 enabled.

Device# monitor capture mycap stop
*Jun 4 14:18:02.503: EPC PROV: Successful. Remove feature object
*Jun 4 14:18:02.504: EPC PROV: Successful. Remove feature object
*Jun 4 14:18:02.504: EPC PROV: Destroyed epc idb subblock
*Jun 4 14:18:02.504: EPC PROV: Found action for policy-map epc_policy_cap1 class-map
epc_class_cap1
*Jun 4 14:18:02.504: EPC PROV: Deleting EPC action
*Jun 4 14:18:02.504: EPC PROV: Successful. CLASS_REMOVE, policy-map epc_policy_cap1, class
 epc_class_cap1
*Jun 4 14:18:02.504: %BUFCAP-6-DISABLE: Capture Point cap1 disabled.
```

CHAPTER 166

Configuring Flexible NetFlow

Prerequisites for Flexible NetFlow

- You are familiar with the Flexible NetFlow key fields as they are defined in the following commands:
  - **match flow**
  - **match interface**
  - **match** {**ipv4** | **ipv6**}
  - **match routing**
  - **match transport**
- You are familiar with the Flexible NetFlow nonkey fields as they are defined in the following commands:
  - **collect counter**
  - **collect flow**
  - **collect interface**
  - **collect**{**ipv4** | **ipv6**}
  - **collect routing**
  - **collect timestamp sys-uptime**
  - **collect transport**
  - **collect policy**
- The networking device must be running a Cisco release that supports Flexible NetFlow.

IPv4 Traffic

- The networking device must be configured for IPv4 routing.
- One of the following must be enabled on your device and on any interfaces on which you want to enable Flexible NetFlow: Cisco Express Forwarding or distributed Cisco Express Forwarding.

IPv6 Traffic

- The networking device must be configured for IPv6 routing.
- One of the following must be enabled on your device and on any interfaces on which you want to enable Flexible NetFlow: Cisco Express Forwarding IPv6 or distributed Cisco Express Forwarding.

Restrictions for Flexible NetFlow

The following are restrictions for Flexible NetFlow:

- Flexible NetFlow is not supported on the Layer 2 port-channel interface, but is supported on the Layer 2 port-channel member ports.
- Flexible NetFlow is supported on the Layer 3 port-channel interfaces and member ports but not on both at the same time for the same traffic type and direction.
- Multiple Flexible Netflow configurations are supported either on port-channel interface or port-channel member ports but not on both.
- Traditional NetFlow accounting is not supported.
- Flexible NetFlow Version 9 and Version 10 export formats are supported. However, if you have not configured the export protocol, Version 9 export format is applied by default.
- For wired Application Visibility and Control (AVC) traffic, only one flow monitor can be configured on one or more Layer 2 or Layer 3 physical interfaces on the system.
- Flexible NetFlow and NBAR cannot be configured together at the same time on the same interface.
- Layer 2, IPv4, and IPv6 traffic types are supported. Multiple flow monitors of different traffic types can be applied for a given interface and direction. Multiple flow monitors of same traffic type cannot be applied for a given interface and direction.
- Layer 2, VLAN, Layer 3 and SVI interfaces are supported, but the device does not support tunnels.
- Depending on the switch type, a switch will have one or two forwarding ASICs. The capacities listed in the above table are on a per-Core/per-ASIC basis.
- The switch has a single core ASIC and the total flows supported is 16K flows (8K per ingress and egress directions). The TCAM limit is 128 entries per direction.
- The NetFlow tables are on separate compartments and cannot be combined. Depending on which core processed the packet, the flows will be created in the table in the corresponding core.
- NetFlow hardware implementation supports four hardware samplers. You can select a sampler rate from 1 out of 2 to 1 out of 1024. Both — random and deterministic — sampling modes are supported.

- NetFlow hardware uses hash tables internally. Hash collisions can occur in the hardware. Therefore, in spite of the internal overflow Content Addressable Memory (CAM), the actual NetFlow table utilization could be about 80 percent.
- Depending on the fields that are used for the flow, a single flow could take two consecutive entries. IPv6 and datalink flows also take two entries. In these situations, the effective usage of NetFlow entries is half the table size, which is separate from the above hash collision limitation.
- The device supports up to 15 flow monitors.
- The NetFlow software implementation supports distributed NetFlow export, so the flows are exported from the same device in which the flow was created.
- Ingress flows are present in the ASIC that first received the packets for the flow. Egress flows are present in the ASIC from which the packets actually left the device set up.
- The reported value for the bytes count field (called “bytes long”) is Layer-2-packet-size—18 bytes. For classic Ethernet traffic (802.3), this will be accurate. For all other Ethernet types, this field will not be accurate. Use the "bytes layer2” field, which always reports the accurate Layer 2 packet size. For information about supported Flexible NetFlow fields, see 'Supported Flexible NetFlow Fields' topic.
- Configuration of IPFIX exporter on an AVC flow monitor is not supported.
- Flexible NetFlow export is not supported on the Ethernet management port, GigabitEthernet 1/1.
- When a flow record has only Source Group Tag (SGT) and Destination Group Tag (DGT) fields (or only either of the two) and if both the values are not applicable, then a flow will still be created with zero values for SGT and DGT. The flow records are expected to include source and destination IP addresses, along with SGT and DGT fields.
- On non-Cisco TrustSec interfaces, an SGT value of zero implies that there is no command header. On Cisco TrustSec interfaces, an SGT value of zero implies an unknown tag.
- When a quality of service (QoS) marked packet is received on an interface which has NetFlow configured in the ingress direction, the QoS value of the packet is captured by the NetFlow collector. However, when the packet is received on an interface which has NetFlow configured in the egress direction and the QoS value has been rewritten on ingress by the switch, the new QoS value of the packet is not captured by the collector.
- For an IPv6 flow monitor, Source Group Tag (SGT) and Destination Group Tag (DGT) fields cannot co-exist with MAC address fields.
- NetFlow records do not support MultiProtocol Label Switching-enabled (MPLS-enabled) interfaces.
- A flow monitor cannot be shared across Layer 3 physical interfaces and logical interfaces (such as, Layer 3 port-channel interface, Layer 3 port-channel member, and switch virtual interface [SVI]), but a flow monitor can be shared across logical interfaces or Layer 3 physical interfaces.
- When Flexible NetFlow and Network Address Translation (NAT) are configured on an interface,
  - Flexible NetFlow will display and export the actual flow details; but not the translated flow details. Application-level gateway (ALG) flow details are not part of the actual flow details that are exported.
  - If the ALG traffic gets translated through the CPU, Flexible NetFlow will display and export the translated flow details for the ALG traffic.

- When configuring a flow record if you include the **match application name** field, the corresponding flow monitor will not support a flow sampler.

Information About Flexible NetFlow

The following sections provide information about Flexible NetFlow.

Flexible NetFlow Overview

Flexible NetFlow uses flows to provide statistics for accounting, network monitoring, and network planning.

A flow is a unidirectional stream of packets that arrives on a source interface and has the same values for the keys. A key is an identified value for a field within the packet. You create a flow using a flow record to define the unique keys for your flow.

The device supports the Flexible NetFlow feature that enables enhanced network anomalies and security detection. Flexible NetFlow allows you to define an optimal flow record for a particular application by selecting the keys from a large collection of predefined fields.

All key values must match for the packet to count in a given flow. A flow might gather other fields of interest, depending on the export record version that you configure. Flows are stored in the Flexible NetFlow cache.

You can export the data that Flexible NetFlow gathers for your flow by using an exporter and export this data to a remote system such as a Flexible NetFlow collector. The Flexible NetFlow collector can use an IPv4 address.

You define the size of the data that you want to collect for a flow using a monitor. The monitor combines the flow record and exporter with the Flexible NetFlow cache information.

The Source Group Tag (SGT) and Destination Group Tag (DGT) fields over Flexible NetFlow are supported for IPv6 traffic.

Original NetFlow and Benefits of Flexible NetFlow

Flexible NetFlow allows the flow to be user defined. The benefits of Flexible NetFlow include:

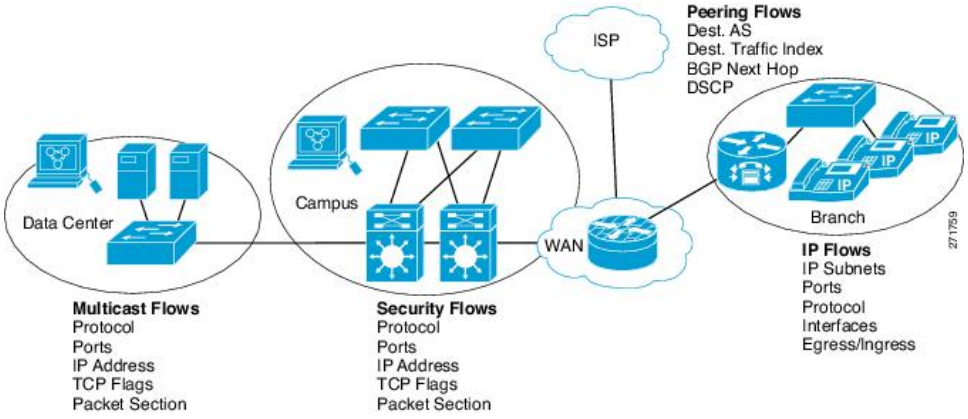
- High-capacity flow recognition, including scalability and aggregation of flow information.
- Enhanced flow infrastructure for security monitoring and dDoS detection and identification.
- New information from packets to adapt flow information to a particular service or operation in the network. The flow information available will be customizable by Flexible NetFlow users.
- Extensive use of Cisco’s flexible and extensible NetFlow Version 9.
- A comprehensive IP accounting feature that can be used to replace many accounting features, such as IP accounting, Border Gateway Protocol (BGP) Policy Accounting, and persistent caches.

Flexible NetFlow allows you to understand network behavior with more efficiency, with specific flow information tailored for various services used in the network. The following are some example applications for a Flexible NetFlow feature:

- Flexible NetFlow enhances Cisco NetFlow as a security monitoring tool. For instance, new flow keys can be defined for packet length or MAC address, allowing users to search for a specific type of attack in the network.
- Flexible NetFlow allows you to quickly identify how much application traffic is being sent between hosts by specifically tracking TCP or UDP applications by the class of service (CoS) in the packets.
- The accounting of traffic entering a Multiprotocol Label Switching (MPLS) or IP core network and its destination for each next hop per class of service. This capability allows the building of an edge-to-edge traffic matrix.

The figure below is an example of how Flexible NetFlow might be deployed in a network.

Figure 152: Typical Deployment for Flexible NetFlow



Flexible NetFlow Components

Flexible NetFlow consists of components that can be used together in several variations to perform traffic analysis and data export. The user-defined flow records and the component structure of Flexible NetFlow facilitates the creation of various configurations for traffic analysis and data export on a networking device with a minimum number of configuration commands. Each flow monitor can have a unique combination of flow record, flow exporter, and cache type. If you change a parameter such as the destination IP address for a flow exporter, it is automatically changed for all the flow monitors that use the flow exporter. The same flow monitor can be used in conjunction with different flow samplers to sample the same type of network traffic at different rates on different interfaces. The following sections provide more information on Flexible NetFlow components:

Flow Records

In Flexible NetFlow a combination of key and nonkey fields is called a record. Flexible NetFlow records are assigned to Flexible NetFlow flow monitors to define the cache that is used for storing flow data. Flexible NetFlow includes several predefined records that can help you get started using Flexible NetFlow.

A flow record defines the keys that Flexible NetFlow uses to identify packets in the flow, as well as other fields of interest that Flexible NetFlow gathers for the flow. You can define a flow record with any combination of keys and fields of interest. The device supports a rich set of keys. A flow record also defines the types of counters gathered per flow. You can configure 64-bit packet or byte counters. The device enables the following match fields as the defaults when you create a flow record:

- **match datalink**—Layer 2 attributes
- **match flow direction**—Specifies a match to the fields identifying the direction of flow.
- **match interface**—Interface attributes
- **match ipv4**—IPv4 attributes
- **match ipv6**—IPv6 attributes
- **match transport**—Transport layer fields
- **match flow cts**—Cisco TrustSec fields

NetFlow Predefined Records

Flexible NetFlow includes several predefined records that you can use to start monitoring traffic in your network. The predefined records are available to help you quickly deploy Flexible NetFlow and are easier to use than user-defined flow records. You can choose from a list of already defined records that may meet the needs for network monitoring. As Flexible NetFlow evolves, popular user-defined flow records will be made available as predefined records to make them easier to implement.

Note Predefined records are not supported for regular Flexible NetFlow.

User-Defined Records

Flexible NetFlow enables you to define your own records for a Flexible NetFlow flow monitor cache by specifying the key and nonkey fields to customize the data collection to your specific requirements. When you define your own records for a Flexible NetFlow flow monitor cache, they are referred to as *user-defined records*. The values in nonkey fields are added to flows to provide additional information about the traffic in the flows. A change in the value of a nonkey field does not create a new flow. In most cases the values for nonkey fields are taken from only the first packet in the flow. Flexible NetFlow enables you to capture counter values such as the number of bytes and packets in a flow as nonkey fields.

You can create user-defined records for applications such as QoS and bandwidth monitoring, application and end user traffic profiling, and security monitoring for dDoS attacks. Flexible NetFlow user-defined records provide the capability to monitor a contiguous section of a packet of a user-configurable size, and use it in a flow record as a key or a nonkey field along with other fields and attributes of the packet. The section may include any Layer 3 data from the packet. The ability to analyze packet fields, enables more detailed traffic monitoring, facilitates the investigation of dDoS attacks, and enables implementation of other security applications such as URL monitoring.

The *bytes* values are the sizes in bytes of these fields in the flow record. If the corresponding fragment of the packet is smaller than the requested section size, Flexible NetFlow will fill the rest of the section field in the flow record with zeros. If the packet type does not match the requested section type, Flexible NetFlow will fill the entire section field in the flow record with zeros.

Flexible NetFlow adds a new Version 9 export format field type for the header and packet section types. Flexible NetFlow will communicate to the NetFlow collector the configured section sizes in the corresponding Version 9 export template fields. The payload sections will have a corresponding length field that can be used to collect the actual size of the collected section.

Flexible NetFlow Match Parameters

The following table describes Flexible NetFlow match parameters. You must configure at least one of the following match parameters for the flow records.

Table 168: Match Parameters

Command	Purpose
match datalink {**dot1q** | **ethertype** | **mac** | **vlan** }	Specifies a match to datalink or Layer 2 fields. The following command options are available: • **dot1q**—Matches to the dot1q field. • **ethertype**—Matches to the ethertype of the packet. • **mac**—Matches the source or destination MAC fields. • **vlan**—Matches to the VLAN that the packet is located on (input or output).
match flow direction	Specifies a match to the flow identifying fields.
match interface {**input** | **output**}	Specifies a match to the interface fields. The following command options are available: • **input**—Matches to the input interface. • **output**—Matches to the output interface.
match ipv4 {**destination** | **protocol** | **source** | **tos** | **ttl** | **version**}	Specifies a match to the IPv4 fields. The following command options are available: • **destination**—Matches to the IPv4 destination address-based fields. • **protocol**—Matches to the IPv4 protocols. • **source**—Matches to the IPv4 source address based fields. • **tos**—Matches to the IPv4 Type of Service fields. • **ttl**—Matches to the IPv4 Time To Live fields. • **version**—Matches to the IP version from the IPv4 header.

Command	Purpose
match ipv6 {**destination** | **hop-limit** | **protocol** | **source** | **traffic-class** | **version** }	Specifies a match to the IPv6 fields. The following command options are available: • **destination**—Matches to the IPv6 destination address-based fields. • **hop-limit**—Matches to the IPv6 hop limit fields. • **protocol**—Matches to the IPv6 payload protocol fields. • **source**—Matches to the IPv6 source address based fields. • **traffic-class**—Matches to the IPv6 traffic class. • **version**—Matches to the IP version from the IPv6 header.
match transport {**destination-port** | **igmp** | **icmp** | **source-port**}	Specifies a match to the Transport Layer fields. The following command options are available: • **destination-port**—Matches to the transport destination port. • **icmp**—Matches to ICMP fields, including ICMP IPv4 and IPv6 fields. • **igmp**—Matches to IGMP fields. • **source-port**—Matches to the transport source port.

Flexible NetFlow Collect Parameters

The following table describes the Flexible NetFlow collect parameters.

Table 169: Collect Parameters

Command	Purpose
collect counter { **bytes** { **layer2** { **long** } | **long** } | **packets** { **long** } }	Collects the counter fields total bytes and total packets.
collect interface {**input** | **output**}	Collects the fields from the input or output interface.
collect timestamp absolute {**first** | **last**}	Collects the fields for the absolute time the first packet was seen or the absolute time the most recent packet was last seen (in milliseconds).

Command	Purpose
collect transport	• **packets**—packet fields • **rcp**—RCP fields • **tcp**—TCP fields
collect transport tcp flags	Collects the following transport TCP flags: • **ack**—TCP acknowledgement flag • **cwr**—TCP congestion window reduced flag • **ece**—TCP ECN echo flag • **fin**—TCP finish flag • **psh**—TCP push flag • **rst**—TCP reset flag • **syn**—TCP synchronize flag • **urg**—TCP urgent flag **Note** On the device, you cannot specify which TCP flag to collect. You can only specify to collect transport TCP flags. All TCP flags will be collected with this command.
collect flow {**sampler** |**username**}	Collects the following flow values: • **sampler**— Configures a flow sampler ID as a non-key field for the record. This field contains the ID of the flow sampler used to monitor the flow. • **username**— Configures the username associated with the flow.
collect ipv4 {**source**|**destination**}	Collects the following IPv4 fields: • **source**— Configures the IPv4 source as a non-key field for a flow record. • **destination**— Configures the IPv4 destination as a non-key field for a flow record.
collect ipv6 {**source**|**destination**}	Collects the following IPv6 information: • **source**— Configures the IPv6 source as a non-key field for a flow record. • **destination**— Configures the IPv6 destination as a non-key field for a flow record.

Command	Purpose
collect routing{ **next-hop address** |**destination**|**source**|**forwarding-status**|**multicast**	Collects the following routing values: • **next-hop address**— Information regarding the next hop from the flows. • **destination**— Destination routing attributes. • **source**— Source routing attributes. • **forwarding status**— Forwarding status of the packet. • **multicast**— Multicast routing attributes.
collect policy firewall event	Configures the collect policy firewall event as a non-key field and enables collecting information on SGACL denied or permitted traffic based on the traffic pattern.

Flow Exporters

Flow exporters export the data in the flow monitor cache to a remote system, such as a server running NetFlow collector, for analysis and storage. Flow exporters are created as separate entities in the configuration. Flow exporters are assigned to flow monitors to provide data export capability for the flow monitors. You can create several flow exporters and assign them to one or more flow monitors to provide several export destinations. You can create one flow exporter and apply it to several flow monitors.

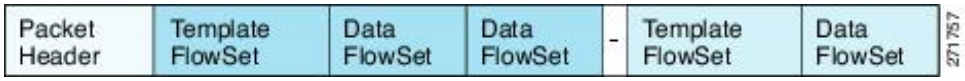
NetFlow Data Export Format Version 9

The basic output of NetFlow is a flow record. Several different formats for flow records have evolved as NetFlow has matured. The most recent evolution of the NetFlow export format is known as Version 9. The distinguishing feature of the NetFlow Version 9 export format is that it is template-based. Templates provide an extensible design to the record format, a feature that should allow future enhancements to NetFlow services without requiring concurrent changes to the basic flow-record format. Using templates provides several key benefits:

- Third-party business partners who produce applications that provide collector or display services for NetFlow do not have to recompile their applications each time a new NetFlow feature is added. Instead, they should be able to use an external data file that documents the known template formats.
- New features can be added to NetFlow quickly without breaking current implementations.
- NetFlow is “future-proofed” against new or developing protocols because the Version 9 format can be adapted to provide support for them.

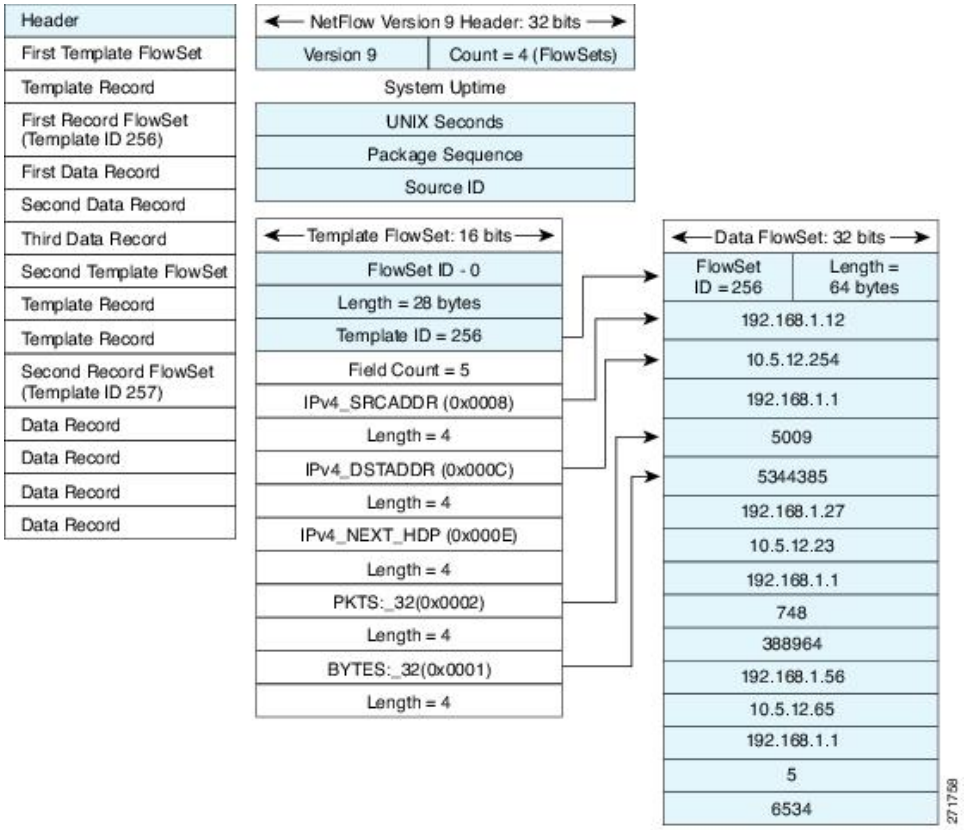
The Version 9 export format consists of a packet header followed by one or more template flow or data flow sets. A template flow set provides a description of the fields that will be present in future data flow sets. These data flow sets may occur later within the same export packet or in subsequent export packets. Template flow and data flow sets can be intermingled within a single export packet, as illustrated in the figure below.

Figure 153: Version 9 Export Packet



NetFlow Version 9 will periodically export the template data so the NetFlow collector will understand what data is to be sent and also export the data flow set for the template. The key advantage to Flexible NetFlow is that the user configures a flow record, which is effectively converted to a Version 9 template and then forwarded to the collector. The figure below is a detailed example of the NetFlow Version 9 export format, including the header, template flow, and data flow sets.

Figure 154: Detailed Example of the NetFlow Version 9 Export Format



For more information on the Version 9 export format, refer to the white paper titled Cisco IOS NetFlow Version 9 Flow-Record Format, available at this URL: http://www.cisco.com/en/US/tech/tk648/tk362/technologies_white_paper09186a00800a3db9.shtml.

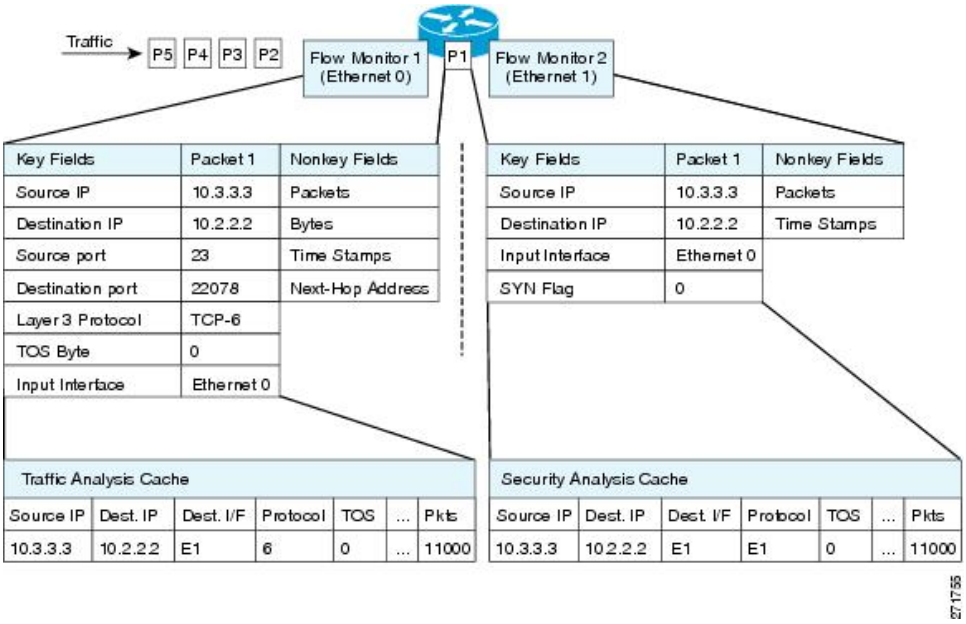
Flow Monitors

Flow monitors are the Flexible NetFlow component that is applied to interfaces to perform network traffic monitoring.

Flow data is collected from the network traffic and added to the flow monitor cache during the monitoring process based on the key and nonkey fields in the flow record.

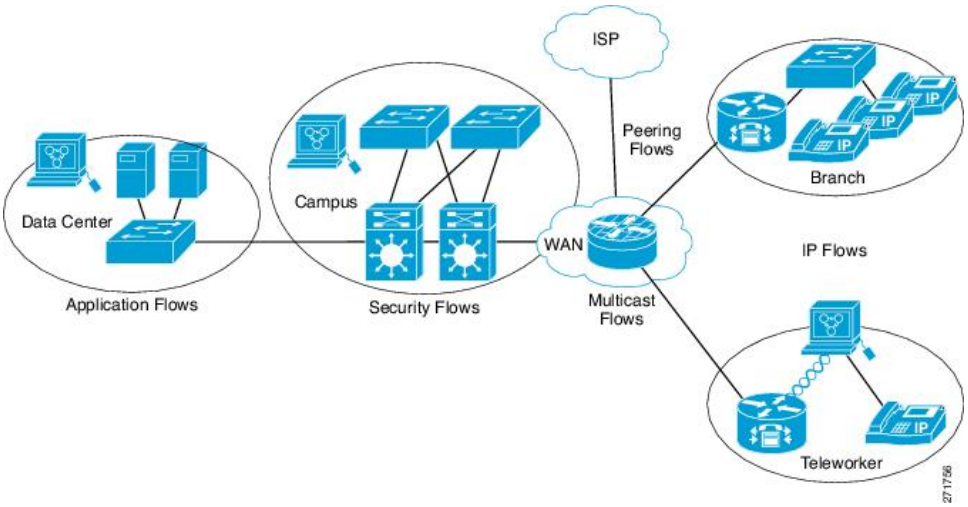
Flexible NetFlow can be used to perform different types of analysis on the same traffic. In the figure below, packet 1 is analyzed using a record designed for standard traffic analysis on the input interface and a record designed for security analysis on the output interface.

Figure 155: Example of Using Two Flow Monitors to Analyze the Same Traffic



The figure below shows a more complex example of how you can apply different types of flow monitors with custom records.

Figure 156: Complex Example of Using Multiple Types of Flow Monitors with Custom Records



Normal

The default cache type is “normal”. In this mode, the entries in the cache are aged out according to the timeout active and timeout inactive settings. When a cache entry is aged out, it is removed from the cache and exported via any exporters configured.

Flow Samplers

Flow samplers are created as separate components in a router's configuration. Flow samplers are used to reduce the load on the device that is running Flexible NetFlow by limiting the number of packets that are selected for analysis.

Flow sampling exchanges monitoring accuracy for router performance. When you apply a sampler to a flow monitor, the overhead load on the router of running the flow monitor is reduced because the number of packets that the flow monitor must analyze is reduced. The reduction in the number of packets that are analyzed by the flow monitor causes a corresponding reduction in the accuracy of the information stored in the flow monitor's cache.

Samplers are combined with flow monitors when they are applied to an interface with the **ip flow monitor** command.

Supported Flexible NetFlow Fields

The following tables provide a consolidated list of supported fields in Flexible NetFlow (FNF) for various traffic types and traffic direction.

Note If the packet has a VLAN field, then that length is not accounted for.

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Key or Collect Fields							
Interface input	Yes	—	Yes	—	Yes	—	If you apply a flow monitor in the input direction: • Use the **match** keyword and use the input interface as a key field. • Use the **collect** keyword and use the output interface as a collect field. This field will be present in the exported records but with a value of 0.

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Interface output	—	Yes	—	Yes	—	Yes	If you apply a flow monitor in the output direction: • Use the **match** keyword and use the output interface as a key field. • Use the **collect** keyword and use the input interface as a collect field. This field will be present in the exported records but with a value of 0.

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Key Fields							
Flow direction	Yes	Yes	Yes	Yes	Yes	Yes	
Ethertype	Yes	Yes	—	—	—	—	
VLAN input	Yes	—	Yes	—	Yes	—	Supported only for a switch port.
VLAN output	—	Yes	—	Yes	—	Yes	Supported only for a switch port.
dot1q VLAN input	Yes	—	Yes	—	Yes	—	Supported only for a switch port.
dot1q VLAN output	—	Yes	—	Yes	—	Yes	Supported only for a switch port.
dot1q priority	Yes	Yes	Yes	Yes	Yes	Yes	Supported only for a switch port.
MAC source address input	Yes	Yes	Yes	Yes	Yes	Yes	

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
MAC source address output	—	—	—	—	—	—	
MAC destination address input	Yes	—	Yes	—	Yes	—	
MAC destination address output	—	Yes	—	Yes	—	Yes	
IPv4 version	—	—	Yes	Yes	Yes	Yes	
IPv4 TOS	—	—	Yes	Yes	Yes	Yes	
IPv4 protocol	—	—	Yes	Yes	Yes	Yes	Must use if any of src/dest port, ICMP code/type, IGMP type or TCP flags are used.
IPv4 TTL	—	—	Yes	Yes	Yes	Yes	
IPv4 TTL	—	—	Yes	Yes	Yes	Yes	Same as IPv4 TTL.
IPv4 protocol	—	—	Yes	Yes	Yes	Yes	Same as IPv4 protocol. Must use if any of src/dest port, ICMP code/type, IGMP type or TCP flags are used.
IPv4 source address	—	—	Yes	Yes	—	—	

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
IPv4 destination address	—	—	Yes	Yes	—	—	
ICMP IPv4 type	—	—	Yes	Yes	—	—	
ICMP IPv4 code	—	—	Yes	Yes	—	—	
IGMP type	—	—	Yes	Yes	—	—	

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Key Fields continued							
IPv6 version	—	—	Yes	Yes	Yes	Yes	Same as IP version.
IPv6 protocol	—	—	Yes	Yes	Yes	Yes	Same as IP protocol. Must use if any of src/dest port, ICMP code/type, IGMP type or TCP flags are used.
IPv6 source address	—	—	—	—	Yes	Yes	
IPv6 destination address	—	—	—	—	Yes	Yes	
IPv6 traffic-class	—	—	Yes	Yes	Yes	Yes	Same as IP TOS.
IPv6 hop-limit	—	—	Yes	Yes	Yes	Yes	Same as IP TTL.
ICMP IPv6 type	—	—	—	—	Yes	Yes	
ICMP IPv6 code	—	—	—	—	Yes	Yes	
source-port	—	—	Yes	Yes	Yes	Yes	

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
dest-port	—	—	Yes	Yes	Yes	Yes	

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Collect Fields							
Bytes long	Yes	Yes	Yes	Yes	Yes	Yes	Packet size = (Ethernet frame size including FCS - 18 bytes) **Recommended:** Avoid this field and use Bytes layer2 long.
Packets long	Yes	Yes	Yes	Yes	Yes	Yes	
Timestamp absolute first	Yes	Yes	Yes	Yes	Yes	Yes	
Timestamp absolute last	Yes	Yes	Yes	Yes	Yes	Yes	
TCP flags	Yes	Yes	Yes	Yes	Yes	Yes	Collects all flags.
Bytes layer2 long	Yes	Yes	Yes	Yes	Yes	Yes	

Default Settings

The following table lists the Flexible NetFlow default settings for the device.

Table 170: Default Flexible NetFlow Settings

Setting	Default
Flow active timeout	1800 seconds
Flow timeout inactive	15 seconds

Flexible NetFlow—Ingress VRF Support Overview

The Flexible NetFlow—Ingress VRF Support feature enables collecting the virtual routing and forwarding (VRF) ID from incoming packets on a device by applying an input flow monitor having a flow record that collects the VRF ID as a key field.

Flexible Netflow—Egress VRF Support Overview

The Flexible Netflow—Egress VRF Support feature enables collecting the VRF ID from outgoing packets on a device by applying an output flow monitor having a flow record that collects the VRF ID as a key field.

Autonomous System Number

The Autonomous System number space is a 32 bit field with 4,294,967,296 unique values, which are available for use to support the Internet's public inter-domain routing system.

An Autonomous System Number (AS number) is a special number assigned by IANA, used primarily with Border Gateway Protocol. It uniquely identifies a network under a single technical administration that has a unique routing policy, or is multi-homed to the public internet. This autonomous system number is required to run BGP and peer with your internet service provider, between internet service providers at peering points, and Internet Exchanges (IX). The AS number must be globally unique so that IP address blocks appear to come from a unique location that BGP can find and route to. BGP uses Prefixes and Autonomous System Paths (AS Paths) to determine the shortest path to a destination where a prefix is located.

NetFlow V9 and IPFIX export types support 32 bit AS number. NetFlow V5 export protocol does not support this 32 AS field, as it follows fixed 16 bit source and destination AS format.

You can export the below BGP parameters in Netflow:

- BGP source origin or peer AS number
- BGP destination origin or peer AS number

Configuration

Use the below command to configure AS number system:

[no] collect routing { **`destination | source`** } as **[[4-octet] peer] [4-octet]**

How to Configure Flexible Netflow

To configure Flexible Netflow, follow these general steps:

1. Create a flow record by specifying keys and non-key fields to the flow.
2. Create an optional flow exporter by specifying the protocol and transport destination port, destination, and other parameters.
3. Create a flow monitor based on the flow record and flow exporter.
4. Create an optional sampler.
5. Apply the flow monitor to a Layer 2 port, Layer 3 port, or VLAN.

Creating a Flow Record

Perform this task to configure a customized flow record.

Customized flow records are used to analyze traffic data for a specific purpose. A customized flow record must have at least one **match** criterion for use as the key field and typically has at least one **collect** criterion for use as a nonkey field.

There are hundreds of possible permutations of customized flow records. This task shows the steps that are used to create one of the possible permutations. Modify the steps in this task as appropriate to create a customized flow record for your requirements.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow record** *record-name* **Example:** `Device(config)# flow record FLOW-RECORD-1`	Creates a flow record and enters Flexible NetFlow flow record configuration mode. • This command also allows you to modify an existing flow record.
Step 4	**description** *description* **Example:** `Device(config-flow-record)# description Used for basic traffic analysis`	(Optional) Creates a description for the flow record.
Step 5	**match** {**ip** | **ipv6**} {**destination** | **source**} **address** **Example:** `Device(config-flow-record)# match ipv4 destination address`	**Note** This example configures the IPv4 destination address as a key field for the record.
Step 6	Repeat Step 5 as required to configure additional key fields for the record.	—
Step 7	**end** **Example:** `Device(config-flow-record)# end`	Exits Flexible NetFlow flow record configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 8	**show flow record** *record-name* **Example:** `Device# show flow record FLOW_RECORD-1`	(Optional) Displays the current status of the specified flow record.
Step 9	**show running-config flow record** *record-name* **Example:** `Device# show running-config flow record FLOW_RECORD-1`	(Optional) Displays the configuration of the specified flow record.

Creating a Flow Exporter

You can create a flow export to define the export parameters for a flow.

Note Each flow exporter supports only one destination. If you want to export the data to multiple destinations, you must configure multiple flow exporters and assign them to the flow monitor.

You can export to a destination using IPv4 address.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**flow exporter** *name* **Example:** `Device(config)# flow exporter ExportTest`	Creates a flow exporter and enters flow exporter configuration mode.
Step 4	**description** *string* **Example:** `Device(config-flow-exporter)#`	(Optional) Describes this flow record as a maximum 63-character string.

	Command or Action	Purpose
	`description ExportV9`	
Step 5	**destination** {*ipv4-address*} **Example:** `Device(config-flow-exporter)# destination 192.0.2.1 (IPv4 destination)`	Sets the IPv4 destination address or hostname for this exporter.
Step 6	**dscp** *value* **Example:** `Device(config-flow-exporter)# dscp 0`	(Optional) Specifies the differentiated services codepoint value. The range is from 0 to 63. The default is 0.
Step 7	**source** *interface type interface number* **Example:** `Device(config-flow-exporter)# source gigabitEthernet 1/1`	(Optional) Specifies the interface to use to reach the NetFlow collector at the configured destination. **Note** Flow Exporter does not support unnumbered IP interface as source interface. The following interfaces can be configured as source: • **Auto Template**—Auto-Template interface • **Capwap**—CAPWAP tunnel interface • **GigabitEthernet**—Gigabit Ethernet IEEE 802 • **GroupVI**—Group virtual interface • **Internal Interface**—Internal interface • **Loopback**—Loopback interface • **Null**—Null interface • **Port-channel**—Ethernet Channel of interface • **Tunnel**—Tunnel interface • **Vlan**—VLANs
Step 8	**transport udp** *number* **Example:**	(Optional) Specifies the UDP port to use to reach the NetFlow collector.

	Command or Action	Purpose
	`Device(config-flow-exporter)# transport udp 200`	
Step 9	**ttl** *seconds* **Example:** `Device(config-flow-exporter)# ttl 210`	(Optional) Configures the time-to-live (TTL) value for datagrams sent by the exporter. The range is from 1 to 255 seconds. The default is 255.
Step 10	**export-protocol** {**netflow-v9**} **Example:** `Device(config-flow-exporter)# export-protocol netflow-v9`	Specifies the version of the NetFlow export protocol used by the exporter.
Step 11	**end** **Example:** `Device(config-flow-record)# end`	Returns to privileged EXEC mode.
Step 12	**show flow exporter** [**name** *record-name*] **Example:** `Device# show flow exporter ExportTest`	(Optional) Displays information about NetFlow flow exporters.
Step 13	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

Define a flow monitor based on the flow record and flow exporter.

Creating a Customized Flow Monitor

Perform this required task to create a customized flow monitor.

Each flow monitor has a separate cache assigned to it. Each flow monitor requires a record to define the contents and layout of its cache entries. These record formats can be one of the predefined formats or a user-defined format. An advanced user can create a customized format using the **flow record** command.

Before you begin

If you want to use a customized record instead of using one of the Flexible NetFlow predefined records, you must create the customized record before you can perform this task. If you want to add a flow exporter to the flow monitor for data export, you must create the exporter before you can complete this task.

Note You must use the **no ip flow monitor** command to remove a flow monitor from all of the interfaces to which you have applied it before you can modify the parameters for the **record** command on the flow monitor.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow monitor** *monitor-name* **Example:** `Device(config)# flow monitor FLOW-MONITOR-1`	Creates a flow monitor and enters Flexible NetFlow flow monitor configuration mode. • This command also allows you to modify an existing flow monitor.
Step 4	**description** *description* **Example:** `Device(config-flow-monitor)# description Used for basic ipv4 traffic analysis`	(Optional) Creates a description for the flow monitor.
Step 5	**record** {*record-name* | **netflow-original** | **netflow** {**ipv4** | **ipv6**} *record* [**peer**]} **Example:** `Device(config-flow-monitor)# record FLOW-RECORD-1`	Specifies the record for the flow monitor.
Step 6	**cache** {**timeout** {**active** | **inactive** | **update**} *seconds* | **type normal** } **Example:** `Device(config-flow-monitor)# cache type normal`	(Optional) Modifies the flow monitor cache parameters such as timeout values, and the cache type. Associates a flow cache with the specified flow monitor.

	Command or Action	Purpose
	`Device(config-flow-monitor)# cache timeout active`	
Step 7	Repeat Step 6 as required to finish modifying the cache parameters for this flow monitor.	—
Step 8	**statistics packet protocol** **Example:** `Device(config-flow-monitor)# statistics packet protocol`	(Optional) Enables the collection of protocol distribution statistics for Flexible NetFlow monitors.
Step 9	**statistics packet size** **Example:** `Device(config-flow-monitor)# statistics packet size`	(Optional) Enables the collection of size distribution statistics for Flexible NetFlow monitors.
Step 10	**exporter** *exporter-name* **Example:** `Device(config-flow-monitor)# exporter EXPORTER-1`	(Optional) Specifies the name of an exporter that was created previously.
Step 11	**end** **Example:** `Device(config-flow-monitor)# end`	Exits Flexible NetFlow flow monitor configuration mode and returns to privileged EXEC mode.
Step 12	**show flow monitor** [[**name**] *monitor-name* [**cache** [**format** {**csv** | **record** | **table**}]] [**statistics**]] **Example:** `Device# show flow monitor FLOW-MONITOR-2 cache`	(Optional) Displays the status and statistics for a Flexible NetFlow flow monitor.
Step 13	**show running-config flow monitor** *monitor-name* **Example:** `Device# show running-config flow monitor FLOW_MONITOR-1`	(Optional) Displays the configuration of the specified flow monitor.
Step 14	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Creating a Flow Sampler

Perform this required task to configure and enable a flow sampler.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**sampler** *sampler-name* **Example:** `Device(config)# sampler SAMPLER-1`	Creates a sampler and enters sampler configuration mode. • This command also allows you to modify an existing sampler.
Step 4	**description** *description* **Example:** `Device(config-sampler)# description Sample at 50%`	(Optional) Creates a description for the flow sampler.
Step 5	**mode** {**random**} **1 out-of** *window-size* **Example:** `Device(config-sampler)# mode random 1 out-of 2`	Specifies the sampler mode and the flow sampler window size. • The range for the *window-size* argument is from 0 to 1024.
Step 6	**exit** **Example:** `Device(config-sampler)# exit`	Exits sampler configuration mode and returns to global configuration mode.
Step 7	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/1`	Specifies an interface and enters interface configuration mode.
Step 8	{**ip** | **ipv6**} **flow monitor** *monitor-name* [[**sampler**] *sampler-name*] {**input** | **output**} **Example:**	Assigns the flow monitor and the flow sampler that you created to the interface to enable sampling.

	Command or Action	Purpose
	`Device(config-if)# ip flow monitor FLOW-MONITOR-1 sampler SAMPLER-1 input`	
Step 9	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 10	**show sampler sampler-name** **Example:** `Device# show sampler SAMPLER-1`	Displays the status and statistics of the flow sampler that you configured and enabled.

Applying a Flow to an Interface

You can apply a flow monitor and an optional sampler to an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type* **Example:** `Device(config)# interface gigabitethernet 1/1`	Enters interface configuration mode and configures an interface. Flexible NetFlow is not supported on the L2 port-channel interface, but is supported on the L2 port-channel member ports. Flexible NetFlow is supported on the L3 port-channel interfaces and member ports but not on both at the same time.
Step 4	{**ip flow monitor** | **ipv6 flow monitor** | **datalink flow monitor**} *name* [**sampler** *name*] {**input** | **output**} **Example:** `Device(config-if)# ip flow monitor MonitorTest input`	Associates an IPv4, IPv6 and datalink flow monitor, and an optional sampler to the interface for input or output packets. **ip flow monitor** – Enables Flexible NetFlow to monitor IPv4 traffic. **ipv6 flow monitor** – Enables Flexible NetFlow to monitor IPv6 traffic.

	Command or Action	Purpose
		datalink flow monitor – Enables Flexible NetFlow to monitor non-IP traffic.
Step 5	**end** **Example:** `Device(config-flow-monitor)# end`	Returns to privileged EXEC mode.
Step 6	**show flow interface** [*interface-type number*] **Example:** `Device# show flow interface`	(Optional) Displays information about NetFlow on an interface.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Bridged NetFlow on a VLAN

You can apply a flow monitor and an optional sampler to a VLAN.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**vlan** [**configuration**] *vlan-id* **Example:** `Device(config)# vlan configuration 30` `Device(config-vlan-config)#`	Enters VLAN or VLAN configuration mode.
Step 4	**ip flow monitor** *monitor name* [**sampler** *sampler name*] {**input** }	Associates a flow monitor and an optional sampler to the VLAN for input packets.

	Command or Action	Purpose
	Example: `Device(config-vlan-config)# ip flow monitor MonitorTest input`	
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Layer 2 NetFlow

You can define Layer 2 keys in Flexible NetFlow records that you can use to capture flows in Layer 2 interfaces.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**flow record** *name* **Example:** `Device(config)# flow record L2_record` `Device(config-flow-record)#`	Enters flow record configuration mode.
Step 4	**match datalink** {**dot1q** |**ethertype** | **mac** | **vlan**} **Example:** `Device(config-flow-record)# match datalink ethertype`	Specifies the Layer 2 attribute as a key.
Step 5	**end** **Example:** `Device(config-flow-record)#  end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show flow record** [*name*] **Example:** `Device# show flow record`	(Optional) Displays information about NetFlow on an interface.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring Flexible NetFlow

The commands in the following table can be used to monitor Flexible NetFlow.

Table 171: Flexible NetFlow Monitoring Commands

Command	Purpose
show flow exporter [**broker** | **export-ids** | **name** | *name* | **statistics** | **templates**]	Displays information about NetFlow flow exporters and statistics.
show flow exporter [**name** *exporter-name*]	Displays information about NetFlow flow exporters and statistics.
show flow interface	Displays information about NetFlow interfaces.
show flow monitor [**name** *exporter-name*]	Displays information about NetFlow flow monitors and statistics.
show flow monitor statistics	Displays the statistics for the flow monitor
show flow monitor cache format {**table** | **record** | **csv**}	Displays the contents of the cache for the flow monitor, in the format specified.
show flow record [**name** *record-name*]	Displays information about NetFlow flow records.
show sampler [**broker** | **name** | *name*]	Displays information about NetFlow samplers.

Configuration Examples for Flexible NetFlow

Example: Configuring a Flow

This example shows how to create a flow and apply it to an interface:

```
Device# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.

Device(config)# flow export export1
Device(config-flow-exporter)# destination 10.0.101.254
Device(config-flow-exporter)# transport udp 2055
Device(config-flow-exporter)# exit
Device(config)# flow record record1
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# match ipv4 protocol
Device(config-flow-record)# match transport source-port
Device(config-flow-record)# match transport destination-port

Device(config-flow-record)# collect counter byte long
Device(config-flow-record)# collect counter packet long
Device(config-flow-record)# collect timestamp absolute first
Device(config-flow-record)# collect timestamp absolute last
Device(config-flow-record)# exit
Device(config)# flow monitor monitor1
Device(config-flow-monitor)# record record1
Device(config-flow-monitor)# exporter export1
Device(config-flow-monitor)# exit
Device(config)# interface gigabitethernet 1/1
Device(config-if)# ip flow monitor monitor1 input
Device(config-if)# end
```

Example: Monitoring IPv4 ingress traffic

This example shows how to monitor IPv4 ingress traffic .

```
Device# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)# flow record  fr-1
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# match interface input
Device(config-flow-record)# collect counter bytes long
Device(config-flow-record)# collect counter packets long
Device(config-flow-record)# collect timestamp absolute first
Device(config-flow-record)# collect timestamp absolute last
Device(config-flow-record)# collect counter bytes layer2 long
Device(config-flow-record)# exit

Device(config)# flow exporter fe-ipfix6
Device(config-flow-exporter)# destination 2001:0:0:24::10
Device(config-flow-exporter)# source Vlan106
Device(config-flow-exporter)# transport udp 4739
Device(config-flow-exporter)# export-protocol ipfix
Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow exporter fe-ipfix
Device(config-flow-exporter)# description IPFIX format collector 100.0.0.80
Device(config-flow-exporter)# destination 100.0.0.80
Device(config-flow-exporter)# dscp 30
```

```
Device(config-flow-exporter)# ttl 210
Device(config-flow-exporter)# transport udp 4739
Device(config-flow-exporter)# export-protocol ipfix
Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow exporter fe-1
Device(config-flow-exporter)# destination 10.5.120.16
Device(config-flow-exporter)# source Vlan105
Device(config-flow-exporter)# dscp 32
Device(config-flow-exporter)# ttl 200
Device(config-flow-exporter)# transport udp 2055

Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow monitor fm-1
Device(config-flow-monitor)# exporter fe-ipfix6
Device(config-flow-monitor)# exporter fe-ipfix
Device(config-flow-monitor)# exporter fe-1
Device(config-flow-monitor)# cache timeout inactive 60
Device(config-flow-monitor)#  cache timeout active 180
Device(config-flow-monitor)# record fr-1
Device(config-flow-monitor)# end

Device# show running-config interface gigabitethernet 1/1
Device# show flow monitor fm-1 cache format table
```

Example: Monitoring IPv4 egress traffic

```
Device# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)# flow record  fr-1 out
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# match interface output
Device(config-flow-record)# collect counter bytes long
Device(config-flow-record)# collect counter packets long
Device(config-flow-record)# collect timestamp absolute first
Device(config-flow-record)# collect timestamp absolute last
Device(config-flow-record)# exit

Device(config)# flow exporter fe-1
Device(config-flow-exporter)# destination 10.5.120.16
Device(config-flow-exporter)# source Vlan105
Device(config-flow-exporter)# dscp 32
Device(config-flow-exporter)# ttl 200
Device(config-flow-exporter)# transport udp 2055
Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow exporter fe-ipfix6
Device(config-flow-exporter)# destination 2001:0:0:24::10
Device(config-flow-exporter)# source Vlan106
Device(config-flow-exporter)# transport udp 4739
Device(config-flow-exporter)# export-protocol ipfix
Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit
```

```
Device(config)# flow exporter fe-ipfix
Device(config-flow-exporter)# description IPFIX format collector 100.0.0.80
Device(config-flow-exporter)# destination 100.0.0.80
Device(config-flow-exporter)# dscp 30
Device(config-flow-exporter)# ttl 210
Device(config-flow-exporter)# transport udp 4739
Device(config-flow-exporter)# export-protocol ipfix
Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow monitor fm-1-output
Device(config-flow-monitor)# exporter fe-1
Device(config-flow-monitor)# exporter fe-ipfix6
Device(config-flow-monitor)# exporter fe-ipfix
Device(config-flow-monitor)# cache timeout inactive 50
Device(config-flow-monitor)#  cache timeout active 120
Device(config-flow-monitor)# record fr-1-out
Device(config-flow-monitor)# end

Device# show flow monitor fm-1-output cache format table
```

Example: Configuring Flexible NetFlow for Ingress VRF Support

The following example configures the collection of theVRF ID from incoming packets on a device by applying an input flow monitor having a flow record that collects the VRF ID as a key field.

```
Device> enable
Device# configure terminal
Device(config)# flow record rm_1
Device(config-flow-record)# match routing vrf input
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# collect interface input
Device(config-flow-record)# collect interface output
Device(config-flow-record)# collect counter packets
Device(config-flow-record)# exit

Device(config)# flow monitor mm_1
Device(config-flow-record)# record rm_1
Device(config-flow-record)# exit

Device(config)# interface GigabitEthernet 1/1
Device(config-if)# ip vrf forwarding green
Device(config-if)# ip address 172.16.2.2 255.255.255.252
Device(config-if)# ip flow monitor mm_1 input
Device(config-if)# end
```

Example: Configuring Flexible NetFlow for Egress VRF Support

The following example configures the collection of the VRF ID from outgoing packets on a device by applying an output flow monitor having a flow record that collects the VRF ID as a key field.

```
Device> enable
Device# configure terminal
Device(config)# flow record rm_1
Device(config-flow-record)# match routing vrf intput
```

```
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# collect interface input
Device(config-flow-record)# collect interface output
Device(config-flow-record)# collect counter packets
Device(config-flow-record)# exit

Device(config)# flow monitor mm_1
Device(config-flow-record)# record rm_1
Device(config-flow-record)# exit

Device(config)# interface GigabitEthernet 1/1
Device(config-if)# ip vrf forwarding green
Device(config-if)# ip address 172.16.2.2 255.255.255.252
Device(config-if)# ip flow monitor mm_1 output
Device(config-if)# end
```

PART VIII

IOx

CHAPTER 167

Configure the Network for IOx Applications

Connections from Switch to IOx Applications

The switch has an additional interface known as AppGigabitEthernet1/1, which can be configured as a standard physical interface in trunk mode for connectivity with IOx applications.

Prerequisites for Establishing a Connection Between Switch and Cisco IOx Applications

- Configure a VLAN ID for the AppGigabitEthernet1/1 interface, regardless of trunk mode.

  Ensure that the configured VLAN is not the default VLAN. By default, VLAN 1 serves as the native VLAN and carries untagged packets.

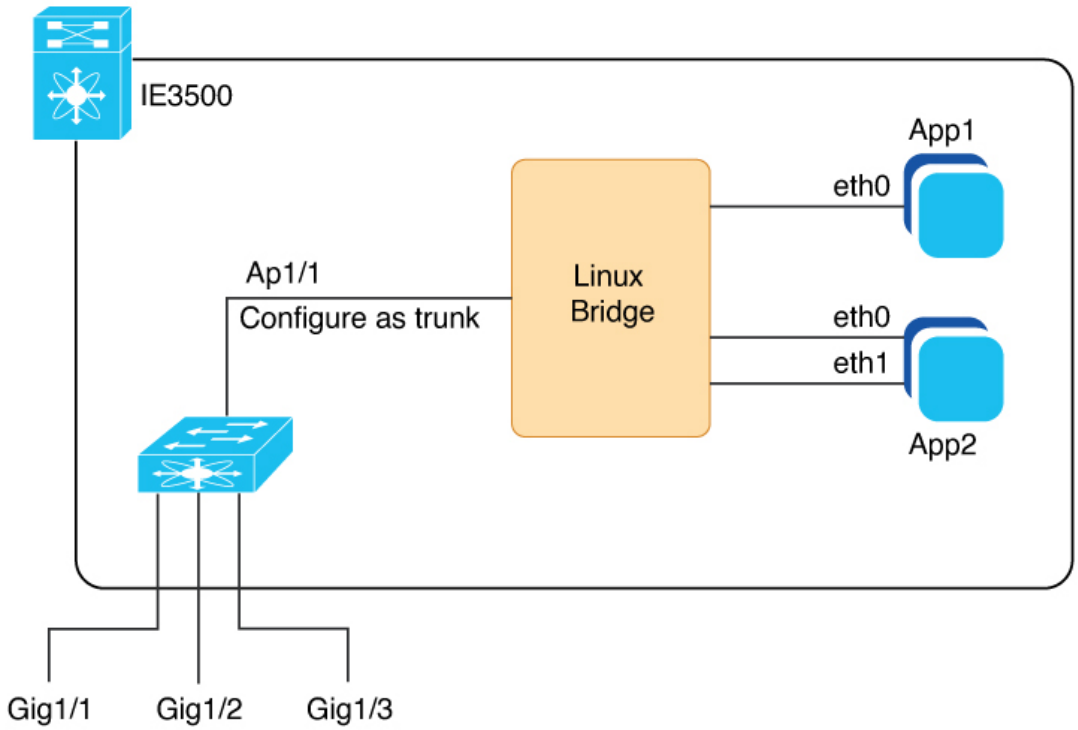
- Assign an IP address to a VLAN interface that is also a member of the AppGigabitEthernet1/1 trunk interface.

Note The VLANs on the AppGigabitEthernet1/1 trunk interface should match VLANs (including management VLANs) that carry data traffic between applications on IOx. For example, if VLAN 10 carries traffic between application and network, including management IP traffic, use the same VLAN for managing applications.

If you intend to use vlan 1 to communicate with the network, then choose a different native VLAN on AppGigabitEthernet1/1.

After configuring the network, enable IOx in the switch. For more information, see Enable IOx Application in the Switch section.

Figure 157: Connections of IOx Network with Applications



This image depicts the Ethernet and Layer 2 connections within an IOx-enabled network on a Switch. The additional interface, AppGigabitEthernet1/1, is internally connected to a Linux bridge and is configured as a trunk to support multiple IOx applications.

The IOx applications can maintain several Ethernet connections and be assigned to any VLAN as required.

See Deployment of IOx Application Using the IOS-XE CLI for an illustration showing a sample network configuration in this guide.

Workflow to Connect and Manage the VLAN

To connect and manage the VLAN, complete all these procedures in given order:

Configure a VLAN for the IOx Interface

Note VLAN ID must be configured on the AppGigabitEthernet1/1 interface, regardless of its trunk mode configuration. Also, verify that the VLAN is routable throughout the network.

Procedure

Step 1 Enter global configuration mode.

```
Device# configure terminal
```

Step 2 Enter interface configuration mode.

```
Device(config)# interface AppGigabitEthernet1/1
```

Step 3 Configure allowed VLANs on the trunk.

```
Device(config-if)#switchport trunk allowed vlan 10
```

Step 4 Configure the interface to operate in trunk mode.

```
Device(config-if)#switchport mode trunk
```

Step 5 Exit interface configuration mode.

```
Device(config-if)#end
```

Configure an SVI address for the VLAN

Procedure

Step 1 Enter global configuration mode.

```
Device# configure terminal
```

Step 2 Enter VLAN interface configuration mode.

```
Device(config)# interface vlan 10
```

Step 3 Assign an IP Address to the VLAN Interface.

```
Device(config-if)#ip address 192.168.0.1 255.255.255.0
```

Step 4 Exit interface configuration mode.

```
Device(config-if)#end
```

Enable IOx Application in the Switch

Before you begin

- Verify that you have a minimum of 4GB memory SD card. Use the SD card for IOx and format it as SD Flash to EXT4, or partition the SD card with at least 1 GB designated for IOs and the remaining space for IOx.
- Verify that you have configured the VLAN for the IOx interface.

Procedure

Step 1 Format the SD card IOx partition with EXT4 filesystem.

```
Device# partition sdflash: iox
Partitioning IOS:IOX(30%:70%) Default
Partition command reloads the switch, Continue?[confirm]
Please make sure to back-up "sdflash:" contents
Partition operation will destroy all data in "sdflash:". Continue? [confirm]
```

Note
The partition command allocates 70 percent of space on the SD card to IOx and 30 percent to IOS as a backup.

Step 2 Enter global configuration mode.

```
Device# configure terminal
```

Step 3 Enable IOx.

```
Device(config)#iox
Warning: Do not remove SD flash card when IOx is enabled or errors on SD device could occur.
*Feb 21 12:49:18.310: %UICFGEXP-6-SERVER_NOTIFIED_START: R0/0: psd: Server iox has been
notified
to start
*Feb 21 12:49:48.165: %IM-6-IOX_ENABLEMENT: R0/0: ioxman: IOX is ready.
```

Step 4 Save the configuration and returns to privileged EXEC mode.

```
Device(config)# end
```

Verify the IOx Infrastructure

Verify that the IOx infrastructure is ready to use, as shown here.

```
Device#show iox-service
IOx Infrastructure Summary:
---------------------------
IOx service (CAF) : Running
IOx service (HA) : Running
IOx service (IOxman) : Running
IOx service (Sec storage) : Running
Libvirtd 5.5.0 : Running
Dockerd v19.03.13-ce : Running
```

CHAPTER 168

IOx Applications Deployment on the Switch

Introduction to IOx Application Management on the Switch

Switch support both LXC and Docker-based applications that utilize ARM64 architecture, offering a range of deployment options. The switches are designed to accommodate IPv4 and IPv6 configurations, allowing for flexible network integration.

The IOx application framework provides configuration options for Docker runtime settings and supports configuring multiple guest or Layer 2 interfaces (ranging from 0 to 63) for each application. Each Layer 2 interface can be assigned to a distinct VLAN, enhancing network organization and segmentation.

Guidelines for IOx Applications Deployment

- Place the application package or tar file in the flash or SD card storage within the IOS partition.
- Use interface AppGigabitEthernet1/1 on the switch to forward Layer 2 application traffic. Verify that the interface is active and configured as a trunk port.
- Use interface AppGigabitEthernet1/1 on the switch to configure Layer 2 interfaces and assign a VLAN with an IP address within the same VLAN network. Next, configure gateway interfaces with an SVI or an IP address in the same network.

Limitations for IOx Application Deployment

- Installing apps on system flash is not allowed.

Methods of IOx Applications Deployment

The Switch supports these two methods for deploying IOx applications.

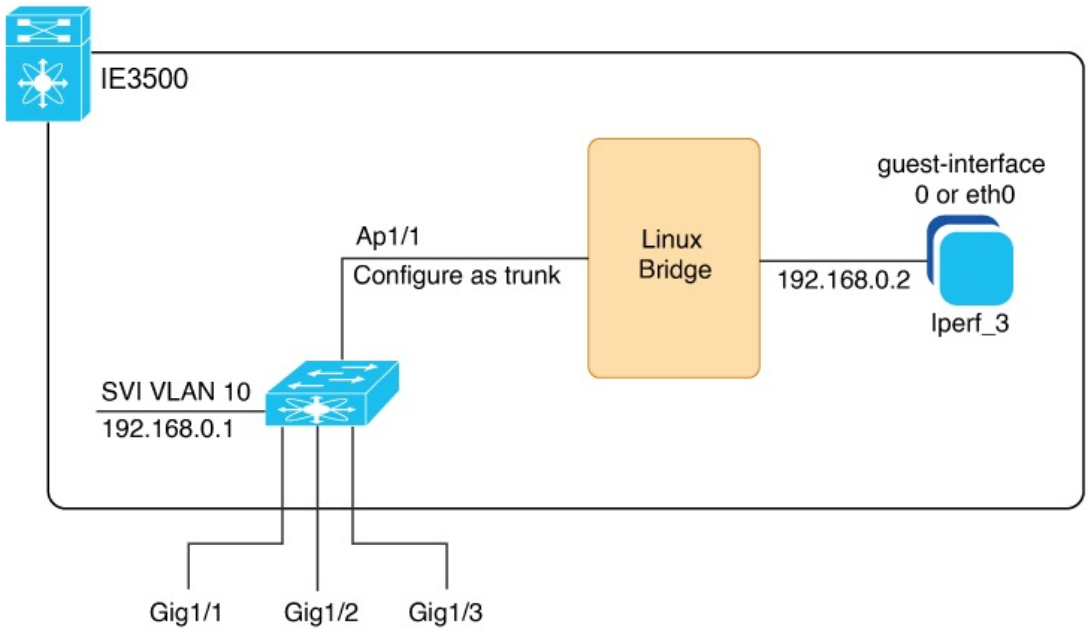
- Deploy IOx application using IOS-XE Command-Line Interface (CLI).
- Deploy IOx application using Cisco IOx Local Manager (GUI).

Resource Profile Options in Cisco IOx Local Manager

- The Cisco IOx Local Manager provides several resource profiles, such as:
  - tiny
  - exclusive
  - default, and
  - custom.
- The custom profile allows you to adjust CPU, memory, and disk allocations according to the specific requirements of your IOx application.

Deployment of IOx Application Using the IOS-XE CLI

Figure 158: Example of IOx Deployment with Application



The configuration example here depicts a typical IOx application deployment on a Switch. The interface AppGigabitEthernet1/1 is internally linked to a Linux bridge and set up as a trunk to facilitate multiple IOx applications. The application "Iperf_3" is assigned the IP address 192.168.0.2 on its guest interface. And the default gateway for the network is configured on the Switch Virtual Interface (SVI) VLAN 10, using the IP address 192.168.0.1.

See Connections from Switch to IOx Applications , on page 2429 for an illustration without interface examples in this guide.

Configure IOx Application Using CLI

Before you begin

Verify that you have configured the network for IOx, as described in the Connections from Switch to IOx Applications , on page 2429 section.

Procedure

Step 1 Enter global configuration mode.

```
Device# configure terminal
```

Step 2 Configure an application name and enter application-hosting configuration mode.

```
Device(config)# app-hosting appid iperf_3
```

Step 3 Configure AppGigabitEthernet trunk.

```
Device(config-app-hosting)#app-vnic AppGigabitEthernet trunk
```

Step 4 Configure a VLAN guest interface. This configuration places Eth0 into VLAN 10.

```
Device(config-config-app-hosting-trunk)#vlan 10 guest-interface 0
```

Step 5 Configure a static IP address.

- IPv4

  ```
  Device(config-config-app-hosting-vlan-access-ip)#guest-ipaddress 192.168.0.2 netmask
  255.255.255.0
  ```

- IPv6

  ```
  Device(config-config-app-hosting-vlan-access-ip)#guest-ipv6address 2001::1 prefix 64
  ```

Step 6 Exit sub-interface mode.

```
Device(config-config-app-hosting-vlan-access-ip)#exit
```

Step 7 Exit app hosting trunk sub-interface mode.

```
Device(config-config-app-hosting-trunk)#exit
```

Step 8 Configure the default gateway for the application. Use the VLAN ID interface of the switch as the gateway.

```
Device(config-app-hosting)#app-default-gateway 192.168.0.1 guest-interface 0
```

Note
Support on default gateway for Ipv4 and one for IPv6.

Step 9 Save the configuration and return to privileged EXEC mode

```
Device(config-if)#end
```

Configure Docker Runtime Options for IOx Applications

Before you begin

- **Set Up Runtime Options:** You can configure up to 30 separate lines of Docker runtime options for IOx applications. The system compiles these options into a single string, proceeding from line 1 through line 30. Each string may contain multiple Docker runtime options.
- **Apply Changes to Runtime Options:** To apply changes to the runtime options, first stop the application, then deactivate it, reactivate it, and finally restart it. This process guarantees the correct implementation of the new runtime options.

Procedure

Step 1 Enter global configuration mode.

```
Device# configure terminal
```

Step 2 Configure an application name and enter application-hosting configuration mode.

```
Device(config)# app-hosting appid iperf_3
```

Step 3 Enter application-hosting Docker-configuration mode.

```
Device(config-app-hosting)#app-resource docker
```

Step 4 Specify the Docker run time options.

```
Device(config-app-hosting-docker)#run-opts 1 "--entrypoint '/bin/sleep 10000'"
```

Step 5 Exit application-hosting Docker-configuration mode.

```
Device(config-app-hosting-docker)#exit
```

Step 6 Save the configuration and return to privileged EXEC mode

```
Device(config-if)#end
```

Configure Application Resource Profiles for Application Hosting

Before you begin

- Before making resource changes, ensure the application is either not installed or in the deployed state.
- Check the memory and storage using **show app-hosting resource** command.

Procedure

Step 1 Enter global configuration mode.

```
Device# configure terminal
```

Step 2 Configure an application name to enter application-hosting configuration mode.

```
Device(config)# app-hosting appid iperf_3
```

Step 3 Configure the custom application resource profile.

```
Device(config-app-hosting)#app-resource profile custom
```

Note
The system supports only custom profile name.

Step 4 Configure the CPU resources.

```
Device(config-app-resource-profile-custom)#cpu 500
```

Step 5 Allocate memory for the application in megabytes.

```
Device(config-app-resource-profile-custom)#memory 256
```

Step 6 Assign persistent disk space for the application, in megabytes.

```
Device(config-app-resource-profile-custom)#persist-disk 256
```

Step 7 Save the configuration and return to privileged EXEC mode

```
Device(config-if)#end
```

Install, Activate, and Start the IOx Application on the Switch

Before you begin

Verify that you have configured the network and the IOx application.

Procedure

Step 1 Install the application and move it into the deployed state.

```
Device#app-hosting install appid iperf_3 package flash:iperf_3_eft_dockerimage_aarch.tar
Installing package 'flash:iperf_3_eft_dockerimage_aarch.tar' for 'iperf_3'. Use 'show
app-hosting list' for progress
```

Note
During installation, the application's signature is verified if signature verification is enabled, which is the default behavior, as described in the Cisco IOx Application Signature Verification and Automatic Activation section.

Step 2 (Optional) Enter this show command to check the state of the IOx application.

```
switch #show app-hosting list
App id                                   State
---------------------------------------------------------
iperf_3                                  DEPLOYED
```

Step 3 Allocate resources and activate the application.

```
Device# app-hosting activate appid iperf_3
Current state is: ACTIVATED
```

Step 4 Start the IOx application.

```
Device# app-hosting start appid iperf_3
iperf_3 started successfully
Current state is: RUNNING
```

Cisco IOx Application Signature Verification and Automatic Activation

IOx infrastructure checks the signature verification of a Cisco IOx application during its installation. The application package signature ensures the validity of the package and confirms that the installed application on the device comes from a trusted source.

Conditions for Signature Verification

IOx infrastructure checks for a signature under these circumstances:

- when signature verification is enabled.
- the application utilizes a restricted resource, such as secure storage.

If signature verification is enabled, and the application lacks a signature, the system prevents the application from running.

Conditions to Run Unsigned Non-Cisco Applications

The system does not permit non-Cisco applications to operate without enabling signature verification. However, the system permits unsigned non-Cisco applications to run if:

- signature verification is disabled.
- the application is not using a restricted resource.

Automated Activation and Startup of Applications with the Start Keyword

The system provides a start keyword option under the application-hosting configuration. When this start keyword is used, the IOx infrastructure automatically activates and starts the application after installation. If the start keyword is not used, manual activation and startup are required using the activate and start commands.

Signature Verification Management and Status Check

Procedure

Step 1 To enable signature verification, use this command.

```
Device#app-hosting verification enable
```

Step 2 (Optional)To disable signature verification, use this command.

```
Device#app-hosting verification disable
```

Step 3 (Optional)To check whether signature verification is enabled or disabled, use this command.

```
Device# show app-hosting infra
IOX version: 2.13.0.0
App signature verification: disabled
Internal working directory: /mnt/usb0/iox
Application Interface Mapping
AppGigabitEthernet Port # Interface Name Port Type Bandwidth
1 AppGigabitEthernet1/1 KR Port - Internal 1G
CPU:
Quota: 33(Percentage)
Available: 26(Percentage)
Quota: 1000(Units)
Available: 800(Units)
```

Note
You can enable or disable sign verification at any time regardless of any installed application states.

Display Maximum Resource Allocation for Application

To display the maximum resources allocated to an application in the switch, use this command:

```
Device# show app-hosting resource
CPU:
  Quota: 33(Percentage)
  Available: 29(Percentage)
VCPU:
  Count: 2
Memory:
  Quota: 4096(MB)
  Available: 3996(MB)
```

```
Storage space:
  Total: 14868(MB)
  Available: 12383(MB)
```

Resources Available in the Switch After IOx Application Configuration

To view the resources remaining in the switch after IOx application configuration, use this command:

```
Device# show app-hosting resource
CPU:
  Quota: 33(Percentage)
  Available: 29(Percentage)
VCPU:
  Count: 2
Memory:
  Quota: 4096(MB)
  Available: 3996(MB)
Storage space:
  Total: 14868(MB)
  Available: 12383(MB)
------------------------------------------
Device# show app-hosting infra
IOX version: 2.13.0.2
App signature verification: enabled
CAF Health: Stable
Internal working directory: /mnt/usb0/iox

Application Interface Mapping
AppGigabitEthernet Port #  Interface Name                Port Type            Bandwidth
           1               AppGigabitEthernet1/1         KR Port - Internal   1G


CPU:
  Quota: 33(Percentage)
  Available: 33(Percentage)
  Quota: 1000(Units)
  Available: 1000(Units)
```

Display Application Information in the Switch

To display detailed application-related information in the switch, use this command:

```
Device# show app-hosting detail
App id                 : iperf
Owner                  : iox
State                  : RUNNING
Application
  Type                 : docker
  Name                 : iperf_3
  Version              : latest
  Description          :
  Author               :
  Path                 : flash:iperf_Vlatest_signed.tar
  URL Path             :
Activated profile name : custom
Resource reservation
  Memory               : 100 MB
  Disk                 : 1000 MB
  CPU                  : 100 units
  CPU-percent          : 10 %
  VCPU                 : 2
```

```
Platform resource profiles
  Profile Name                  CPU(unit)  Memory(MB)  Disk(MB)
  ----------------------------------------------------------------
Attached devices
  Type              Name               Alias
  ---------------------------------------------
  serial/shell     iox_console_shell   serial0
  serial/aux       iox_console_aux     serial1
  serial/syslog    iox_syslog          serial2
  serial/trace     iox_trace           serial3
Network interfaces
   --------------------------------------
eth1:
   eth1:
   MAC address         : 52:54:dd:2e:47:24
   IPv4 address        : <ipv4 address>
   IPv6 address        : <ipv6 address>
   Network name        : mgmt-bridge-v10
   Multicast           : No
   Mirroring           : No
eth0:
   MAC address         : 52:54:dd:d2:ea:de
   IPv6 address        : ::
   Network name        : mgmt-bridge300
   Multicast           : No
   Mirroring           : No

Docker
------
Run-time information
  Command             :
  Entry-point         : /bin/sleep 10000
  Run options in use  : --entrypoint '/bin/sleep 10000'
  Package run options  :
Application health information
  Status              : 0
  Last probe error    :
  Last probe output   :
Device#
```

Stop, Deactivate, and Uninstall IOx Application on the Switch

Procedure

Step 1 To stop the IOx application, use this command.

```
Device# app-hosting stop appid iperf_3
iperf_3 stopped successfully
Current state is: STOPPED
```

Step 2 To deactivate the IOx application, use this command.

```
Device# app-hosting deactivate appid iperf_3
iperf_3 deactivated successfully
Current state is: DEPLOYED
```

Step 3 To uninstall the IOx application, use this command.

```
Device# app-hosting uninstall appid iperf_3
Uninstalling 'iperf_3'. Use 'show app-hosting list' for progress.
```

Display App-Hosting Commands

To display the list of subcommands for the **app-hosting** command, use the command as given here:

```
Device# app-hosting ?
  activate      Application activate  <== to activate app
  clear         Clear console/aux connection     <== to clear console or aux session if
connected
  connect       Application connect              <== to connect the app console or aux or
session once in run state
  data          Application data            <== to upload files to the apps
  deactivate    Application deactivate      <== to deactivate an app
  debug         debug            <== for caf related debug commands
  install       Application install         <== to install app
  move          Move File             <== to move trace or core file
  settings      Application settings        <== to configure app specific setting using
file
  start         Application start           <== to start an app
  stop          Application stop            <== to stop an app
  uninstall     Application uninstall       <== to uninstall an app`
  upgrade       Application upgrade         <== to upgrade app to new version
  verification Application signature verification setting (global)  <== to enable/disable
the sign verification
```

Deploy an IOx Application using Cisco IOx Local Manager

Cisco IOx Local Manager offers a web-based interface for managing, administering, monitoring, and troubleshooting applications on a host system and to perform various related activities.

You can access Cisco IOx Local Manager from the web-based UI and use Cisco IOx Local Manager to deploy applications.

Switch Configuration

1. Enable the web server.

   ```
   Device(config)# ip http secure-server
   ```

2. Create a user account for access.

   ```
   Device(config)# username admin privilege 15 password 0 secret
   ```

Access the Cisco IOx Local Manager Application

1. Log in to the web-based UI.
2. Navigate to **Configuration > IOx**. The IOx option is located under the **Services** section.
3. In the Cisco IOx Local Manager, enter your Cisco IOS username and password.
4. Click Log In to proceed.

See Cisco IOx Local Manager Reference Guide. page for more information

PART IX

Protocols and Timing

CHAPTER 169

Precision Time Protocol

Precision Time Protocol

Precision Time Protocol (PTP) is defined in IEEE 1588 as Precision Clock Synchronization for Networked Measurements and Control Systems, and was developed to synchronize the clocks in packet-based networks that include distributed device clocks of varying precision and stability. PTP is designed specifically for industrial, networked measurement and control systems, and is optimal for use in distributed systems because it requires minimal bandwidth and little processing overhead.

Benefits of PTP

Smart grid power automation applications such as peak-hour billing, virtual power generators, and outage monitoring and management, require precise time accuracy and stability. Timing precision improves network monitoring accuracy and troubleshooting ability.

In addition to providing time accuracy and synchronization, the PTP message-based protocol can be implemented on packet-based networks, such as Ethernet networks. The benefits of using PTP in an Ethernet network include:

- Low cost and easy setup in existing Ethernet networks
- Limited bandwidth is required for PTP data packets

Message-Based Synchronization

To ensure clock synchronization, PTP requires an accurate measurement of the communication path delay between time source (grandmaster clock) and the time recipient. PTP sends messages between the time source and time recipient to determine the delay measurement. Then, PTP measures the exact message transmit and receive times and uses these times to calculate the communication path delay. PTP then adjusts current time information contained in network data for the calculated delay, resulting in more accurate time information.

This delay measurement principle determines path delay between devices on the network. The local clocks are adjusted for this delay using a series of messages sent between time source and time recipient devices. The one-way delay time is calculated by averaging the path delay of the transmit and receive messages. This calculation assumes a symmetrical communication path; however, switched networks do not necessarily have symmetrical communication paths, due to the buffering process.

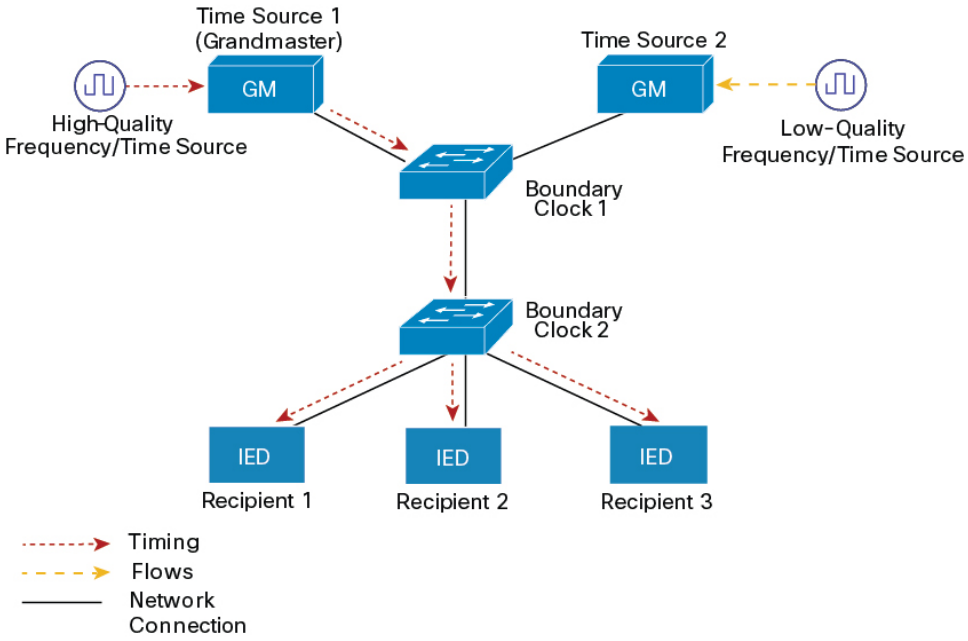
PTP provides a method, using transparent clocks, to measure and account for the delay in a time-interval field in network timing packets. Doing so makes the switches temporarily transparent to the time source and time recipient nodes on the network. An end-to-end transparent clock forwards all messages on the network in the same way that a switch does.

Note Cisco PTP supports multicast PTP messages only.

To read a detailed description of synchronization messages, refer to PTP Event Message Sequences, on page 2449. To learn more about how transparent clocks calculate network delays, refer to Transparent Clock, on page 2453.

The following figure shows a typical 1588 PTP network that includes grandmaster clocks, switches in boundary clock mode, and Intelligent Electronic Device (IEDs) such as a digital relays or protection devices. In this diagram, Time Source 1 is the grandmaster clock. If Time Source 1 becomes unavailable, the time recipient boundary clocks switch to Time Source 2 for synchronization.

Figure 159: PTP Network



PTP Event Message Sequences

This section describes the PTP event message sequences that occur during synchronization.

Synchronizing with Boundary Clocks

The ordinary and boundary clocks configured for the delay request-response mechanism use the following event messages to generate and communicate timing information:

- Sync
- Delay_Req
- Follow_Up
- Delay_Resp

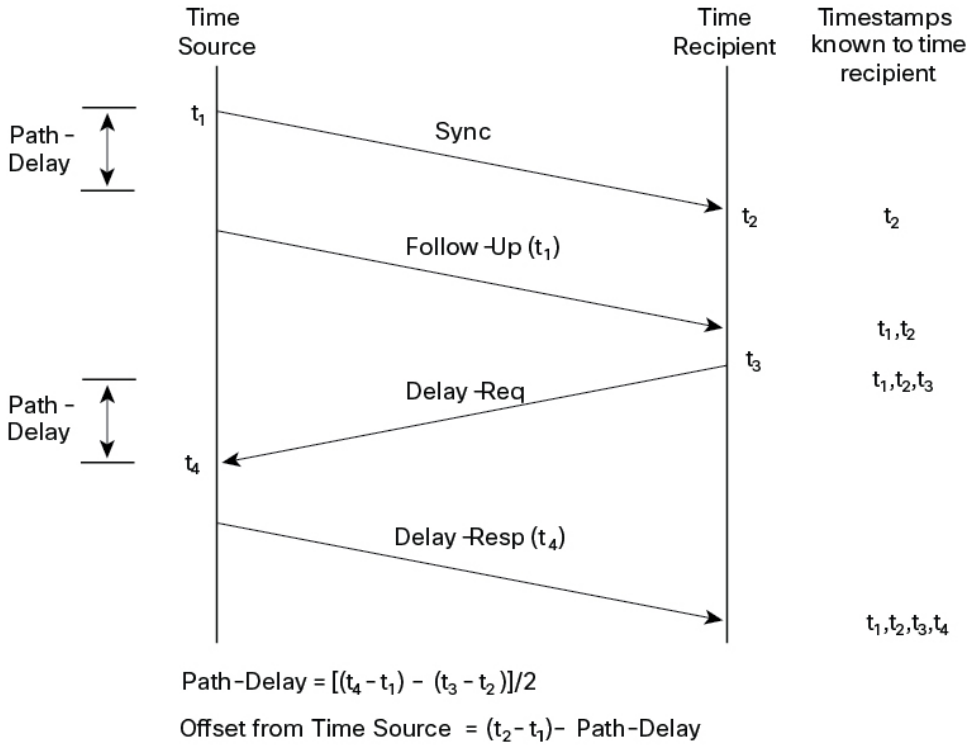
These messages are sent in the following sequence:

1. The time source sends a Sync message to the time recipient and notes the time (t1) at which it was sent.
2. The time recipient receives the Sync message and notes the time of reception (t2).
3. The time source conveys to the time recipient the timestamp t1 by embedding the timestamp t1 in a Follow_Up message.
4. The time recipient sends a Delay_Req message to the time source and notes the time (t3) at which it was sent.
5. The time source receives the Delay_Req message and notes the time of reception (t4).
6. The time source conveys to the time recipient the timestamp t4 by embedding it in a Delay_Resp message.

After this sequence, the time recipient possesses all four timestamps. These timestamps can be used to compute the offset of the time recipient clock relative to the time source, and the mean propagation time of messages between the two clocks.

The offset calculation is based on the assumption that the time for the message to propagate from time source to time recipient is the same as the time required from time recipient to time source. This assumption is not always valid on an Ethernet network due to asymmetrical packet delay times.

Figure 160: Detailed Steps—Boundary Clock Synchronization



Synchronizing with Peer-to-Peer Transparent Clocks

When the network includes multiple levels of boundary clocks in the hierarchy, with non-PTP enabled devices between them, synchronization accuracy decreases.

The round-trip time is assumed to be equal to mean_path_delay/2, however this is not always valid for Ethernet networks. To improve accuracy, the resident time of each intermediary clock is added to the offset in the end-to-end transparent clock. Resident time, however, does not consider the link delay between peers, which is handled by peer-to-peer transparent clocks.

Peer-to-peer transparent clocks measure the link delay between two clock ports implementing the peer delay mechanism. The link delay is used to correct timing information in Sync and Follow_Up messages.

Peer-to-peer transparent clocks use the following event messages:

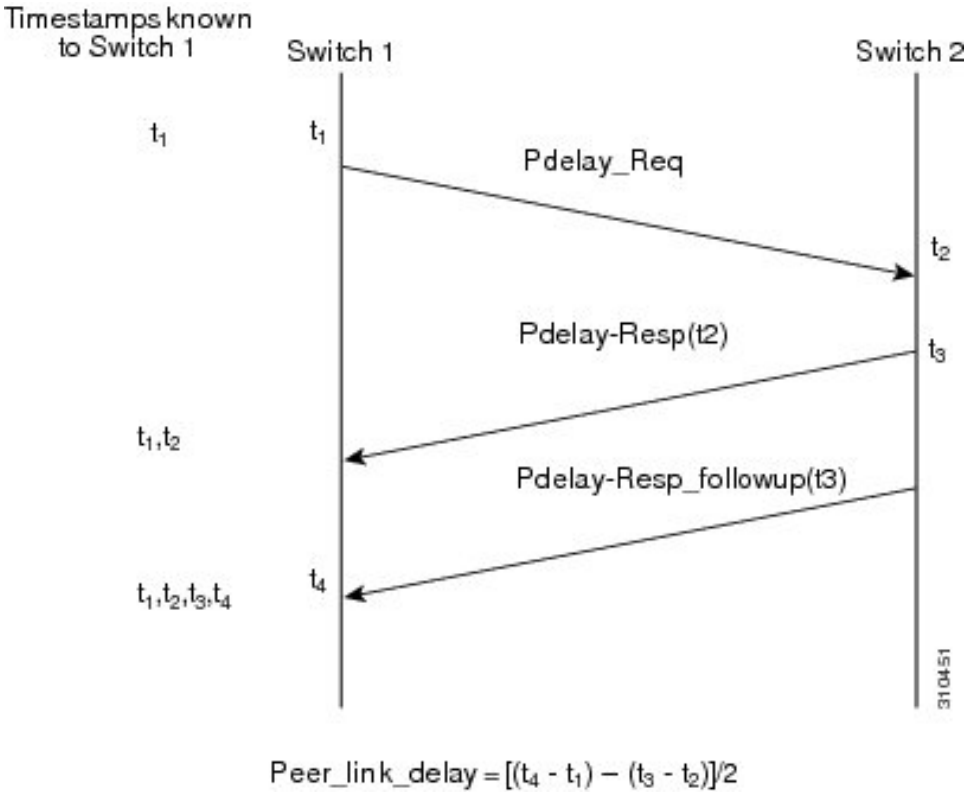
- Pdelay_Req
- Pdelay_Resp
- Pdelay_Resp_Follow_Up

These messages are sent in the following sequence:

1. Port 1 generates timestamp t1 for a Pdelay_Req message.
2. Port 2 receives and generates timestamp t2 for this message.
3. Port 2 returns and generates timestamp t3 for a Pdelay_Resp message.

   To minimize errors due to any frequency offset between the two ports, Port 2 returns the Pdelay_Resp message as quickly as possible after the receipt of the Pdelay_Req message.
4. Port 2 returns timestamps t2 and t3 in the Pdelay_Resp and Pdelay_Resp_Follow_Up messages respectively.
5. Port 1 generates timestamp t4 after receiving the Pdelay_Resp message. Port 1 then uses the four timestamps (t1, t2, t3, and t4) to calculate the mean link delay.

Figure 161: Detailed Steps—Peer-to-Peer Transparent Clock Synchronization



Synchronizing the Local Clock

In an ideal PTP network, the time source and time recipient clocks operate at the same frequency. However, *drift* can occur on the network. Drift is the frequency difference between the time source and time recipient clocks. You can compensate for drift by using the time stamp information in the device hardware and follow-up messages (intercepted by the switch) to adjust the frequency of the local clock to match the frequency of the time source clock.

Best Master Clock Algorithm

The Best Master Clock Algorithm (BMCA) is the basis of PTP functionality. The BMCA specifies how each clock on the network determines the best time source clock in its subdomain of all the clocks it can see, including itself. The BMCA runs on the network continuously and quickly adjusts for changes in network configuration.

The BMCA uses the following criteria to determine the best time source clock in the subdomain:

- Clock quality (for example, GPS is considered the highest quality)
- Clock accuracy of the clock's time base.
- Stability of the local oscillator
- Closest clock to the grandmaster

In addition to identifying the best time source clock, the BMCA also ensures that clock conflicts do not occur on the PTP network by ensuring that:

- Clocks do not have to negotiate with one another.
- There is no misconfiguration, such as two time source clocks or no time source clocks, as a result of the time source clock identification process.

PTP Clocks

A PTP network is made up of PTP-enabled devices and devices that are not using PTP. The PTP-enabled devices typically consist of the following clock types.

Note Transparent Clock mode is the only clock mode supported in Power Profile 2017. See PTP Profiles, on page 2454 in this document.

Grandmaster Clock

The grandmaster clock is a network device physically attached to the server time source. All clocks are synchronized to the grandmaster clock.

Within a PTP domain, the grandmaster clock is the primary source of time for clock synchronization using PTP. The grandmaster clock usually has a precise time source, such as a GPS or atomic clock. When the network does not require any external time reference and only needs to be synchronized internally, the grandmaster clock can free run.

Boundary Clock

A boundary clock in a PTP network operates in place of a standard network switch or router. Boundary clocks have more than one PTP port, and each port provides access to a separate PTP communication path. They intercept and process all PTP messages, and pass all other network traffic. The boundary clock uses the BMCA to select the best clock seen by any port. The selected port is then set to nonmaster mode. The master port synchronizes the clocks connected downstream, while the nonmaster port synchronizes with the upstream master clock.

Transparent Clock

The role of transparent clocks in a PTP network is to update the time-interval field that is part of the PTP event message. This update compensates for switch delay and has an accuracy of within one picosecond.

There are two types of transparent clocks:

End-to-end (E2E) transparent clocks measure the PTP event message transit time (also known as *resident time)* for SYNC and DELAY_REQUEST messages. This measured transit time is added to a data field (correction field) in the corresponding messages:

- The measured transit time of a SYNC message is added to the correction field of the corresponding SYNC or the FOLLOW_UP message.
- The measured transit time of a DELAY_REQUEST message is added to the correction field of the corresponding DELAY_RESPONSE message.

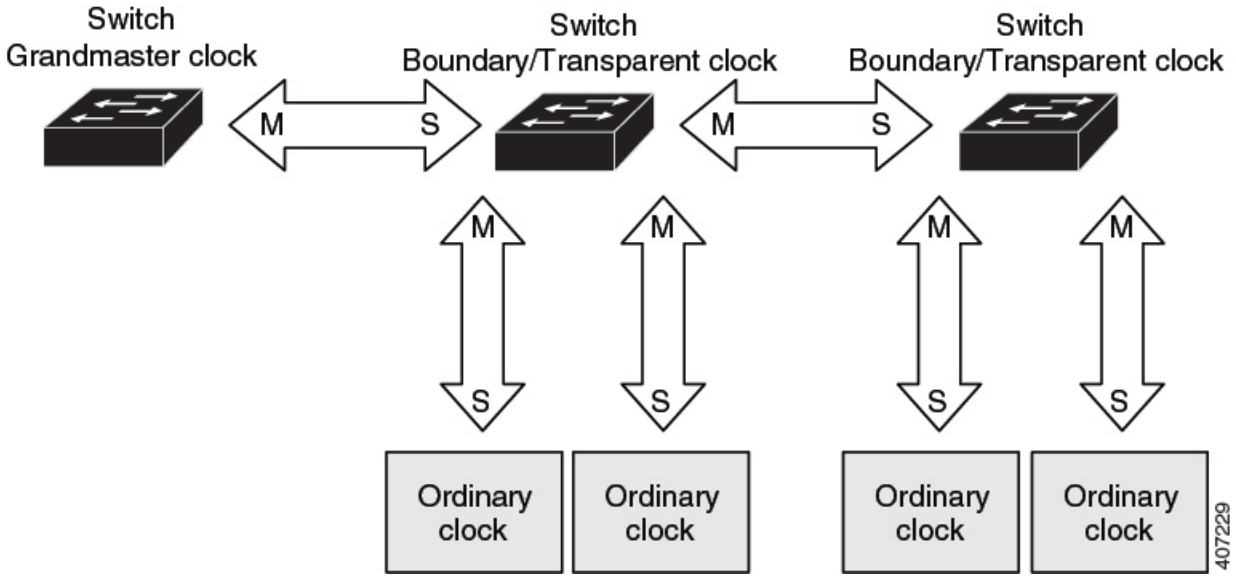
The time recipient uses this information when determining the offset between the time recipient's and the time source's time. E2E transparent clocks do not provide correction for the propagation delay of the link itself.

Peer-to-peer (P2P) transparent clocks measure PTP event message transit time in the same way E2E transparent clocks do, as described above. In addition, P2P transparent clocks measure the upstream link delay. The upstream link delay is the estimated packet propagation delay between the upstream neighbor P2P transparent clock and the P2P transparent clock under consideration.

These two times (message transit time and upstream link delay time) are both added to the correction field of the PTP event message, and the correction field of the message received by the time recipient contains the sum of all link delays. In theory, this is the total end-to-end delay (from time source to time recipient) of the SYNC packet.

The following figure illustrates PTP clocks in a time source-time recipient hierarchy within a PTP network.

Figure 162: PTP Clock Hierarchy



Note In the preceding illustration, *M* signifies master port, and *S* signifies nonmaster, or subordinate port.

Clock Configuration

- All PHY PTP clocks are synchronized to the grandmaster clock. The switch system clock is not synchronized as part of PTP configuration and processes.
- When VLAN is enabled on the grandmaster clock, it must be in the same VLAN as the native VLAN of the PTP port on the switch.
- Grandmaster clocks can drop untagged PTP messages when a VLAN is configured on the grandmaster clock. To force the switch to send tagged packets to the grandmaster clock, enter the global **vlan dot1q tag native** command.

PTP Profiles

This section describes the following PTP profiles available on the switch:

- Default Profile
- Power Profile

Power Profile-2011 is supported as defined in PC37.238-2011 - IEEE Draft Standard Profile for Use of IEEE 1588 Precision Time Protocol in Power System Applications. This documentation uses the terms Power Profile mode and Default Profile mode when referring to this IEEE 1588 profile and its associated configuration values.

Two Power Profiles are supported: Power Profile-2011 and Power Profile-2017. Power Profile-2017 is defined in IEEE Standard C37.238™-2017 (Revision of IEEE Std C37.238-2011) for use of IEEE 1588 Precision Time Protocol in Power System Applications.

This documentation uses the terms Power Profile mode and Default Profile mode when referring to this IEEE 1588 profile and its associated configuration values. The IEEE 1588 definition of a PTP profile is *the set of allowed PTP features applicable to a device*. A PTP profile is usually specific to a particular type of application or environment and defines the following values:

- Best master clock algorithm options
- Configuration management options
- Path delay mechanisms (peer delay or delay request-response)
- Range and default values of all PTP configurable attributes and data set members
- Transport mechanisms that are required, permitted, or prohibited
- Node types that are required, permitted, or prohibited
- Options that are required, permitted, or prohibited

Default Profile Mode

The default PTP profile mode on the switch is Default Profile mode. In this mode:

- Supports transparent clock, boundary clock, grandmaster boundary clock, and PTP forward mode (PTP passthrough) on the default profile.
- Ordinary clocks are not supported.

Power Profile Mode

The IEEE Power Profile defines specific or allowed values for PTP networks used in power substations. The defined values include the optimum physical layer, the higher-level protocol for PTP messages, and the preferred best master clock algorithm. The Power Profile values ensure consistent and reliable network time distribution within substations, between substations, and across wide geographic areas.

The following table lists the configuration values defined by the IEEE 1588 Power Profile and the values that the switch uses for each PTP profile mode.

Table 172: Configuration Values for the IEEE PTP Power Profile and Switch Modes

PTP Field	Switch Configuration Value	
	Power Profile Mode	**Default Profile Mode**
Message transmission	**Access ports**: Untagged Layer 2 packets. **Trunk ports**: PTP packets are tagged with the PTP VLAN. If the PTP VLAN is not configured, packets go untagged over the native VLAN.	Layer 3 packets. By default, 802.1q tagging is disabled.
MAC address– Nonpeer delay messages	01-00-5e-00-01-81.	Default profile uses L3 transport multicast address 224.0.1.129 for all PTP messages. Equivalent mac address is 01-00-5e-00-01-81.
MAC address– Peer delay messages	01-80-C2-00-00-0E.	Not applicable to this mode.
Domain number	0.	0.
Path delay calculation	Peer-to-peer transparent clocks using the peer_delay mechanism.	End-to-end transparent clocks using the delay_request mechanism.
BMCA	Enabled.	Enabled.
Clock type	Two-step.	Two-step.
Time scale	Epoch.	Epoch.
Grandmaster ID and local time determination	PTP-specific TLV to indicate Grandmaster ID.	PTP-specific type, length, and value to indicate Grandmaster ID.
Time accuracy over network hops	Over 16 hops, end device synchronization accuracy is within 1 usec (1 microsecond).	Not applicable in this mode.

PTP Profile Comparison

Table 173: Comparison of PTP Profiles on IE Switches

Profile	Default (*)		Power Profile-2011		Power Profile-2017
Standard	IEEE1588 v2 (J.3)		IEEE C37.238-2011		IEEE C37.238-2017
Mode	Boundary	End-to-End transparent	Boundary	Peer-to-Peer transparent	Peer-to-Peer transparent
Path Delay	Delay req/res	Delay req/res	Peer delay req/res	Peer delay req/res	Peer delay req/res
Non-PTP device allowed in PTP domain	Yes	Yes	No	No	No
Transport	UDP over IP (multicast)		L2 Multicast		L2 Multicast

* Delay Request-Response Default PTP profile (as defined in IEEE1588 J.3).

Tagging Behavior of PTP Packets

The following table describes the switch tagging behavior in Power Profile and Default Profile modes.

Table 174: Tagging Behavior for PTP Packets

Switch Port Mode	Configuration	Power Profile Mode		Default Profile Mode	
		Behavior	Priority	Behavior	Priority
Trunk Port	**vlan dot1q tag native** enabled	Switch tags packets	7	Switch tags packets	7
Trunk Port	**vlan dot1q tag native** disabled	PTP software tags packets	4	Untagged	None
Access Port	N/A	Untagged	None	Untagged	None

Configurable Boundary Clock Synchronization Algorithm

You can configure the BC synchronization algorithm to accommodate various PTP use cases, depending on whether you need to prioritize filtering of input time errors or faster convergence. A PTP algorithm that filters packet delay variation (PDV) converges more slowly than a PTP algorithm that does not.

By default, the BC uses a linear feedback controller (that is, a servo) to set the BC's time output to the next clock. The linear servo provides a small amount of PDV filtering and converges in an average amount of time. For improved convergence time, BCs can use the TC feedforward algorithm to measure the delay added by the network elements forwarding plane (the disturbance) and use that measured delay to control the time output.

While the feedforward BC dramatically speeds up the boundary clock, the feedforward BC does not filter any PDV. The adaptive PDV filter provides high-quality time synchronization in the presence of PDV over wireless access points (APs) and enterprise switches that do not support PTP and that add significant PDV.

Three options are available for BC synchronization (all are compliant with IEEE 1588-2008):

- Feedforward: For very fast and accurate convergence; no PDV filtering.
- Adaptive: Filters as much PDV as possible, given a set of assumptions about the PDV characteristics, the hardware configuration, and the environmental conditions.

Note With the adaptive filter, the switch does not meet the time performance requirements specified in ITU-T G.8261.

- Linear: Provides simple linear filtering (the default).

Adaptive mode (**ptp transfer filter adaptive**) is not available in Power Profile mode.

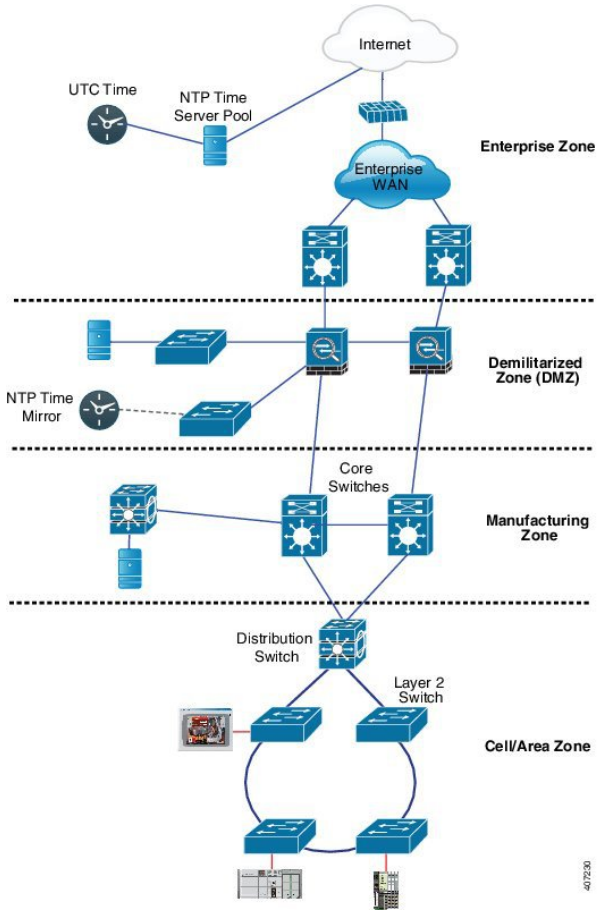
NTP to PTP Time Conversion

NTP to PTP Time Conversion allows you to use Network Time Protocol (NTP) as a time source for PTP. Customers who use PTP for precise synchronization within a site can use NTP across sites, where precise synchronization is not required.

NTP is the traditional method of synchronizing clocks across packet-based networks. NTP uses a two-way time transfer mechanism, between a time source and an end device. NTP is capable of synchronizing a device within a few 100 milliseconds across the Internet, and within a few milliseconds in a tightly controlled LAN. The ability to use NTP as a time source for PTP allows customers to correlate data generated in their PTP network with data in their enterprise data centers running NTP.

The following figure shows an example of an industrial network based on the Industrial Automation and Control System Reference Model. The enterprise zone and demilitarized zone run NTP, and the manufacturing zone and cell/area zone run PTP with NTP as the time source. The switch with the NTP to PTP conversion feature can be either the Layer 2 Switch or the Distribution Switch in the Cell/Area Zone.

Figure 163: Industrial Network with NTP and PTP



Note The NTP to PTP feature supports the Default E2E Profile and Power Profile.

Clock Manager

The clock manager is the component in the Cisco NTP to PTP software architecture that tracks the various time services and selects the clock that actively provides time. The clock manager notifies the time services of important changes, such as state changes, leap seconds, or daylight saving time.

The clock manager selects the NTP or manually set clock first, followed by PTP and the real-time clock if NTP is not active. The following table shows the results of the clock selection process.

Table 175: Time Service Selection

NTP (Active) or Manually Set	PTP (Active)	Real-Time Clock	Selected Output
True	Don’t care	Don’t care	NTP or Manually Set

NTP (Active) or Manually Set	PTP (Active)	Real-Time Clock	Selected Output
False	True	Don’t care	PTP
False	False	True	Real-Time Clock

In general, the clock manager ensures that the time displayed in the Cisco IOS commands **show ptp clock** and **show clock** match. The **show clock** command always follows this priority, but there are two corner cases where the **show ptp clock** time may differ:

- The switch is either a TC or a BC, and there is no other active reference on the network. To preserve backwards compatibility, the TC and BC never take their time from the clock manager, only from the network PTP GMC. If there is no active PTP GMC, then the time displayed in the **show clock** and the **show ptp clock** command output may differ.
- The switch is a synchronizing TC, a BC with a subordinate port, or a GMC-BC with subordinate port, and the time provided by the PTP GMC does not match the time provided by NTP or the user (that is, manually set). In this case, the PTP clock must forward the time from the PTP GMC. If the PTP clock does not follow the PTP GMC, then the PTP network ends up with two different time bases, which would break any control loops or sequence of event applications using PTP.

The following table shows how the Cisco IOS and PTP clocks behave given the various configurations. Most of the time, the two clocks match. Occasionally, the two clocks are different; those configurations are highlighted in the table.

Table 176: Expected Time Flow

IOS Clock Configuration	PTP Clock Configuration	IOS Clock Source	PTP Clock Source
Calendar	PTP BC, E2E TC, or GMC-BC in BC Mode	PTP	PTP
Manual	PTP BC, E2E TC, or GMC-BC in BC Mode	Manual	PTP
NTP	PTP BC, E2E TC, or GMC-BC in BC Mode	NTP	PTP
Calendar	GMC-BC in GM Mode	Calendar	Calendar
Manual	GMC-BC in GM Mode	Manual	Manual
NTP	GMC-BC in GM Mode	NTP	NTP

GMC Block

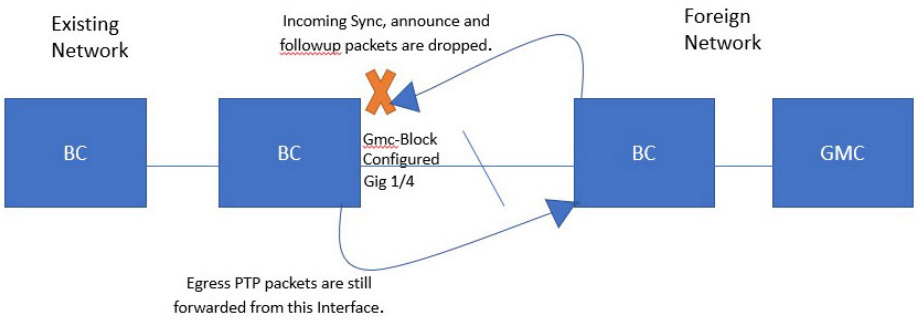
GMC Block protects an existing network from any rogue GMC that might try to synchronize with the devices inside the network. This feature is supported for all PTP clock modes except Forward mode. After the feature is enabled on an interface, only the egress Announce, Sync, and Followup PTP packets are allowed and all ingress Announce, Sync, and Followup packets are dropped on this interface. This prevents the port state transition to time recipient.

Information about a rogue GMC is retrieved from the packets before dropping them. However, egress PTP packets are still allowed from this interface, so it can act as a GMC. To identify the rogue device, details such as IP address and clock ID are stored and displayed for the interface. Two Syslog messages are also generated to notify the presence and clearance of rogue devices.

You can configure PTP gmc-block on multiple ports, if you suspect multiple foreign networks are connected to your existing system. Per-port Syslog messages are displayed after an interval of 30 seconds of receiving rogue packets and after 180 -240 seconds when packets stop coming. Relay minor alarms and SNMP traps are also generated to notify of the presence of foreign rogue devices.

Packet Flow with GMC Block

The following figure shows an example of a PTP network topology with the GMC Block feature configured on an interface.



PTP packets originate in the GMC of the foreign network in an attempt to sync with the existing network. When the PTP packets reach the port configured with GMC Block, the packets are dropped after the system retrieves the required information from them.

Because packets from the foreign network are restricted, the system syncs with the local GMC present in the existing system. PTP packets originated on the port configured with GMC Block are still allowed to egress from this interface, which allows devices in the existing network to be GMC.

Guidelines and Limitations

This section lists the guidelines and limitations when using PTP.

General PTP Guidelines

- The Cisco PTP implementation supports only the two-step clock and not the one-step clock.
- Cisco PTP supports multicast PTP messages only.
- Cisco PTP supports only PTP version2.
- Power Profile-2017 supports only transparent clock mode.

PTP Mode and Profile

- The switch and the grandmaster clock must be in the same PTP domain.
- When Power Profile mode is enabled, the switch drops the PTP announce messages that do not include these two Types, Length, Value (TLV) message extensions: *Organization_extension* and *Alternate_timescale*.

  If the grandmaster clock is not compliant with PTP and sends announce messages without these TLVs, configure the switch to process the announce message by entering the following command:

```
ptp clock boundary domain 1 profile power
allow-without-tlv
```

- When the switch is in Power Profile mode, only the peer_delay mechanism is supported.

  To enable power profile boundary mode and associate interfaces using the clock-port suboption, enter the following command:

  ```
  ptp clock boundary domain 1 profile power
  clock-port 1
  transport ethernet multicast interface gi1/1
  ```

- To disable power profile transparent mode, enter the following command, which returns the switch to forward mode.

  ```
  no ptp clock transparent domain x profile power
  ```

- To enable the E2E transparent clock, use the following command:

  ```
  ptp clock transparent domain x profile default
  ```

- In Default Profile mode, only the delay_request mechanism is supported.

  To enable default profile boundary clock mode and interfaces associated with clock-port suboption, enter the following command:

  ```
  ptp clock boundary domain 1 profile default
  clock-port 1
  transport ipv4 multicast interface gi1/1
  ```

Packet Format

- The packet format for PTP messages can be 802.1q tagged packets or untagged packets.
- The switch does not support 802.1q QinQ tunneling of PTP packets.
- In Power Profile mode:
  - When the PTP interface is configured as an access port, PTP messages are sent as untagged, Layer 2 packets.
  - When the PTP interface is configured as a trunk port, two cases are possible:
    - When native VLAN is enabled on the interface, PTP packets go untagged over the native VLAN.
    - When PTP VLAN is configured under the clock-port, PTP packets are tagged with the PTP VLAN configured.
- Time recipient IEDs must support tagged and untagged packets.
- When PTP packets are sent on the native VLAN in E2E Transparent Clock Mode, they are sent as untagged packets. To configure the switch to send them as tagged packets, enter the global **vlan dot1q tag native** command.

NTP to PTP Conversion

The NTP to PTP feature supports the Default E2E Profile and Power Profile.

PTP Interaction with Other Features

- PTP over Media Redundancy Protocol (MRP) is not supported.
- PTP over Port Channels is not supported.
- PTP over Cisco Resilient Ethernet Protocol (REP) is not supported.
- The following PTP clock modes only operate on a single VLAN:
  - e2etransparent
  - p2ptransparent

Default Settings

- PTP is enabled on the switch by default.
- By default, the switch uses configuration values defined in the Default Profile (Default Profile mode is enabled).
- The switch default PTP clock mode is E2E Transparent Clock Mode.
- The default BC synchronization algorithm is linear filter.

VLAN Configuration

This section contains information about VLAN configuration.

- Sets the PTP VLAN on a trunk port. The range is from 1 to 4094. The default is the native VLAN of the trunk port.
- In boundary mode, only PTP packets in PTP VLAN are processed; PTP packets from other VLANs are dropped.
- Before configuring the PTP VLAN on an interface, the PTP VLAN must be created and allowed on the trunk port.
- Most grandmaster clocks use the default VLAN 0. In Power Profile mode, the switch default VLAN is VLAN 1 and VLAN 0 is reserved. When you change the default grandmaster clock VLAN, it must be changed to a VLAN other than 0.
- When VLAN is disabled on the grandmaster clock, the PTP interface must be configured as an access port.

Configuring GMC Mode

The following sections provide steps for configuring GMC mode for default and power profiles:

- Configuring GMC Mode for a Default Profile, on page 2463
- Configure GMC Mode for a Power Profile, on page 2463

Configuring GMC Mode for a Default Profile

Complete the steps in this section to configure GMC mode for a default profile.

Procedure

	Command or Action	Purpose
Step 1	**ptp clock boundary domain** *domain number* **profile default** **Example:** `switch(config)# ptp clock boundary domain 0 profile default`	Enable the default profile boundary mode.
Step 2	**gmc default** **Example:** `switch(config-ptp-clk)# gmc default`	Enable the GMC boundary clock.
Step 3	**clock-port** *port name* **Example:** `switch(config-ptp-clk)# clock-port port1`	Define a new clock port.
Step 4	**transport ipv4 multicast** *interface type interface number* **Example:** `switch(config-ptp-port)# transport ipv4 multicast interface Gi1/1`	Specify the transport mechanism for clocking traffic.

Configure GMC Mode for a Power Profile

Complete the steps in this section to configure GMC mode for a power profile.

Procedure

	Command or Action	Purpose
Step 1	**ptp clock boundary domain** *domain number* **profile power** **Example:** `switch(config)# ptp clock boundary domain 0 profile power`	Enable the power profile boundary mode.
Step 2	**gmc default** **Example:** `switch(config-ptp-clk)# gmc default`	Enable the GMC boundary clock.

	Command or Action	Purpose
Step 3	**clock-port** *port name* **Example:** `switchswitch(config-ptp-clk)# clock-port port1`	Defines a new clock port.
Step 4	**transport ethernet multicast** *interface type interface number* **Example:** `switch(config-ptp-port)# transport ethernet multicast interface gi1/1`	Specifies the transport mechanism for clocking traffic.

Configuring PTP Default Profile

This section describes how to configure the switch to operate in Default Profile mode.

Configure a Boundary Clock

If an interface is not added as part of BC clock, it will be in forward mode exchanging PTP packets, which will cause instability in PTP operation. To avoid this, it is recommended to disable PTP on all such interfaces using the **no ptp enable** command.

Follow these steps to configure the switch as a boundary clock:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `switch> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `switch# configure terminal`	Enters configuration mode.
Step 3	**ptp clock boundary domain** *domain-number* **profile default** **Example:** `switch(config)# ptp clock boundary domain 0 profile default`	This step configures the boundary type PTP clock, which terminates the PTP session from the grandmaster clock and acts as a PTP server or client clock downstream.
Step 4	**clock-port** *port-name* **Example:** `switch(config-ptp-clk)# clock-port dyn1`	Defines a new clock port.

	Command or Action	Purpose
Step 5	**transport ipv4 multicast interface** *interface-type interface-number* **Example:** `switch(config-ptp-port)# transport ipv4 multicast interface Gi1/1`	Specifies the transport mechanism for clocking traffic.
Step 6	(Optional) **vlan** *vlan-id* **Example:** `config-ptp-port)# vlan 100`	Configure VLAN for tagged packets.

Example

Example of Untagged

```
ptp clock boundary domain 0  profile default
 clock-port dyn1
  transport ipv4 multicast interface Gi1/1
 clock-port dyn2
  transport ipv4 multicast interface Gi1/1
```

Example of Tagged

```
ptp clock boundary domain 0  profile default
 clock-port dyn1
  transport ipv4 multicast interface Gi1/1
  vlan 100
 clock-port dyn2
  transport ipv4 multicast interface Gi1/1
  vlan 200
```

Configure a Transparent Clock

All interfaces will be part of TC mode once configured.

Follow these steps to configure the switch as a transparent clock.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `switch> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `switch# configure terminal`	Enters configuration mode.
Step 3	**ptp clock transparent domain** *domain-number* **profile default**	This step configures the transparent type PTP clock, which updates the PTP time correction

	Command or Action	Purpose
	Example: `switch(config)# ptp clock transparent domain 0  profile default`	field to account for the delay in forwarding the traffic.
Step 4	(Optional) **vlan** *vlan-id* **Example:** `(config-ptp-clk)# vlan 100`	Configure VLAN for tagged packets.

Example

Example of Untagged

```
ptp clock transparent domain 0  profile default
```

Example of Tagged

```
ptp clock transparent domain 0  profile default
vlan 100
```

Configuring a PTP Power Profile

This section describes how to configure the switch to use the PTP Power Profile.

The Power Profile defines a subset of PTP which is intended to run over layer 2 networks, that is, Ethernet, but no Internet Protocol.

Note Power Profile-2017 is supported only in Transparent Clock mode.

Configure a Boundary Clock

If an interface is not added as part of BC clock, it is in forward mode exchanging PTP packets, which causes instability in PTP operation. To avoid this, disable PTP on all such interfaces using the **no ptp enable** command.

Follow these steps to configure the switch as a boundary clock:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters configuration mode.

	Command or Action	Purpose
	`switch# configure terminal`	
Step 3	**ptp clock boundary domain** *domain-number* **profile power** **Example:** `switch(config)# ptp clock boundary domain 0 profile default`	This step configures the boundary type PTP clock, which stops the PTP session from the grandmaster clock and acts as a PTP server or client clock downstream.
Step 4	**clock-port** *port-name* **Example:** `switch(config-ptp-clk)# clock-port dyn1`	Defines a new clock port.
Step 5	**transport ethernet multicast interface** *interface-type interface-number* **Example:** `switch(config-ptp-port)# transport ethernet multicast interface Gi1/1`	Specifies the transport mechanism for clocking traffic.
Step 6	(Optional) **vlan** *vlan-id* **Example:** `(config-ptp-port)# vlan 100`	Configure VLAN for tagged packets.

Example

Example of Untagged

```
ptp clock boundary domain 0  profile power
 clock-port dyn1
  transport ethernet multicast interface Gi1/1
 clock-port dyn2
  transport ethernet multicast interface Gi1/2
```

Example of Tagged

```
ptp clock boundary domain 0  profile power
 clock-port dyn1
  transport ethernet multicast interface Gi1/1
  vlan 100
 clock-port dyn2
  transport ethernet multicast interface Gi1/2
  vlan 100
```

Example of not Including TLV Extensions

```
ptp clock boundary domain 0 profile power
allow-without-tlv
```

Configure a Transparent Clock

Follow these steps to configure the switch as a transparent clock.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `switch# configure terminal`	Enters configuration mode.
Step 3	**ptp clock transparent domain** *domain-number* **profile power** **Example:** `switch(config)# ptp clock transparent domain 0  profile power`	This step configures the transparent type PTP clock, which updates the PTP time correction field to account for the delay in forwarding the traffic.
Step 4	• (Power Profile-2011): **ptp clock transparent domain** *domain-number* **profile power** • (Power Profile-2017): **ptp clock transparent domain** *domain-number* **profile power-2017** **Example:** `switch(config)# ptp clock transparent domain 0  profile power` **Example:** `switch(config)# ptp clock transparent domain 254  profile power-2017`	This step configures the transparent type PTP clock, which updates the PTP time correction field to account for the delay in forwarding the traffic. The update helps improve the accuracy of the 1588 clock at the client.
Step 5	(Optional) **vlan** *vlan-id* **Example:** `(config-ptp-clk)# vlan 100`	Configure VLAN for tagged packets.

Example

Example of Untagged

```
ptp clock transparent domain 0 profile power
```

Example of Tagged

```
ptp clock transparent domain 0 profile power
vlan 100
```

Example of Tagged Power Profile-2017

```
ptp clock transparent domain 254 profile power-2017
vlan 100
```

Example of not Including TLV Extensions: Power Profile-2011

```
ptp clock transparent domain 0 profile power
allow-without-tlv
```

Example of not Including TLV Extensions: Power Profile-2017

```
ptp clock transparent domain 0 profile power-2017
allow-without-tlv
```

Enable PTP Forward Mode

Complete the steps in this section to enable PTP forward mode.

To enable PTP forward mode, and remove existing PTP clock configurations, you remove the existing PTP clock. When you do so, all interfaces automatically become part of forward mode.

Note Forward mode supports only the default profile.

Procedure

	Command or Action	Purpose
Step 1	**ptp clock boundary domain** *domain-number* **profile default** **Example:** `switch(config)# ptp clock boundary domain 0 profile default`	Configure the boundary type PTP clock. Doing so terminates the PTP session from the grandmaster clock and acts as a PTP server or client clock downstream.
Step 2	**clock-port** *port-name* **Example:** `switch(config)# clock-port 1`	Define a new clock port.
Step 3	**transport ipv4 multicast interface** *interface-type interface-number* **Example:** `switch(config-ptp-port)# transport ipv4 multicast interface Gi1/1`	Specifies the transport mechanism for clocking traffic.
Step 4	**exit**	Enters global configuration mode.
Step 5	**no ptp clock boundary domain** *domain-number* **profile default** **Example:** `switch(config)# no ptp clock boundary domain 0 profile default`	Remove the PTP clock configuration.
Step 6	**end**	Exit global configuration mode and returns to privileged EXEC mode.

Remove PTP Forward Mode

Complete the steps in this section to remove PTP forward mode.

To remove forward PTP forward mode configuration, you enable a PTP clock.

Note Forward mode supports only the default profile.

Procedure

Step 1 **no ptp clock**

Configure a clock to get out of forward mode.

Step 2 **ptp clock boundary domain** *domain-number* **profile default**

Example:

```
switch(config)# ptp clock boundary domain 0 profile default
```

Configure the boundary type PTP clock. Doing so terminates the PTP session from the grandmaster clock and acts as a PTP server or client clock downstream.

Step 3 **end**

Exit global configuration mode and returns to privileged EXEC mode.

Disable PTP

Complete the steps in this section to disable PTP on an interface.

Note The following procedure applies to both default and power modes.

Procedure

	Command or Action	Purpose
Step 1	**interface** *interface-id* **Example:** `switch(config)# interface gi1/1`	Enter interface configuration mode.
Step 2	**no ptp enable**	Disable PTP on the interface.

Enable GMC Block in Boundary Mode

Complete the steps in this section to enable GMC Block in boundary mode.

Procedure

	Command or Action	Purpose
Step 1	**ptp clock boundary domain** *domain number* **profile default** **Example:** `switch(config)# ptp clock boundary domain 0 profile default`	Configure the boundary type PTP clock, which terminates the PTP session from the grandmaster clock and acts as a PTP server or client clock downstream.
Step 2	**clock-port** *port-name* **Example:** `switch(config-ptp-clk)# clock-port 1`	Define a new clock port.
Step 3	**transport ipv4 multicast interface** *interface type interface number* **Example:** `switch(config-ptp-port)# transport ipv4 multicast interface Gi1/1`	
Step 4	**gmc-block** **Example:** `switch(config-ptp-port)# gmc-block`	Enable GMC Block.

Enable GMC Block in Transparent Mode

Complete the steps in this section to enable GMC Block in transparent mode.

Procedure

	Command or Action	Purpose
Step 1	**ptp clock transparent domain** *domain number* **profile power** **Example:** `switch(config)# ptp clock transparent domain 0 profile power`	This step configures the transparent type PTP clock, which updates the PTP time correction field to account for the delay in forwarding the traffic. The transparent clock can update some fields in the PTP packets to ensure that the client has greater time accuracy.
Step 2	**gmc-block** *interface* **Example:**	Enable GMC Block.

	Command or Action	Purpose
	`switch(config-ptp-clk)# gmc-block gi1/1`	

PTP Alarms

PTP alarms can help you manage and monitor PTP on the switch. You can configure the PTP alarms to trigger the external alarm relay output and send system messages to a syslog server. The PTP alarms are raised only once for the first 5-minute interval and subsequently once every 30 minutes. PTP alarms are disabled by default.

The following sequence describes how PTP alarm timing works:

1. PTP alarm monitoring starts 5 minutes after bootup.
2. The PTP alarm is raised only once for the first 5-minute interval and subsequently once for an interval of 30 minutes.
3. The alarms are damped when there is continuous state change, for example, PTP port state flapping or PTP parent flapping.

The following table describes the types of PTP alarms:

Table 177: PTP Alarms

Alarm	Alarm Type	Clock Mode Supported	Description
PTP SLAVE port state change	Minor	Boundary and transparent clock modes	This alarm is raised when the PTP port state changes from “SLAVE” to any of the following PTP port states: Initializing, Faulty, Disabled, Listening, Pre_Master, Master, Passive, or Uncalibrated. A system message is generated when the PTP port state transitions between Slave and Passive Slave. This alarm remains raised until you clear the alarm.

Alarm	Alarm Type	Clock Mode Supported	Description
PTP PASSIVE_SLAVE port state change	Minor	Boundary and transparent clock modes	This alarm is raised when the PTP port state changes from "PASSIVE-SLAVE" to any of the following PTP port states: Initializing, Faulty, Disabled, Listening, Pre_Master, Master, Passive, or Uncalibrated. A system message is generated when the PTP port state transitions between Slave and Passive Slave.
PTP Parent change	Minor	Boundary clock mode	This alarm raised when there is a change in PTP parent. This alarm remains raised until you clear the alarm.
PTP Time Property Clock Synchronized	Minor	Transparent clock mode	This alarm is raised when the PTP Clock Time Property "Clock Syntonized" field changes from TRUE to FALSE. This alarm is cleared when the "Clock Syntonized" field changes from FALSE to TRUE.

Configuring PTP Alarms

To enable and configure the global PTP alarms:

Procedure

Step 1 Enter global configuration mode:

configure terminal

Step 2 Enable PTP alarms:

alarm facility ptp enable

Step 3 Enable notifications to be sent to an SNMP server:

alarm facility ptp notifies

Step 4 Associate the PTP alarms to a relay.

alarm facility ptp relay major

Step 5 Send PTP alarm traps to a syslog server.

alarm facility ptp syslog

Example

```
Switch#configure terminal
Switch(config)#alarm facility ptp enable
Switch(config)#alarm facility ptp syslog
Switch(config)#end
Switch#show alarm settings
…..
…..
…..
PTP
     Alarm     Enabled
     Relay     MIN
     Notifies  Enabled
     Syslog    Enabled
Switch#show facility-alarm status
Source                  Severity Description                          Relay    Time
Switch                  MINOR    32 PTP Clock Parent change           NONE     Mar 09 2022
 01:23:45
GigabitEthernet1/1    MINOR    5 PTP SLAVE port state changed      NONE     Mar 09 2022
01:23:45
```

SNMP Support for PTP MIBs

SNMP management information bases (MIBs) for Precision Time Protocol (PTP) is supported. These include CISCO-PTP-MIB. The feature enables you to get PTP-related information from a switch remotely.

The MIB is supported with boundary clock and transparent clock modes. It is supported in both the default and power profiles.

SNMP is an application-layer protocol that provides a message format for communication between SNMP managers and agents. SNMP provides a standardized framework and a common language used for monitoring and managing devices in a network. An SNMP networks includes the following components:

- **SNMP Manager:** A system that controls and monitors the activities of network hosts using SNMP. The most common managing system is a network management system (NMS). The term An can be a dedicated device used for network management or the applications used on such a device.
- **SNMP Agent:** The software component within a managed device that maintains the data for the device and reports this data, as needed, to managing systems. The agent resides on the switch. To enable an SNMP agent on a Cisco switch, you must define the relationship between the manager and the agent.
- **SNMP MIB**: An SNMP agent contains MIB variables. The SNMP manager can request information from an agent to store information in the agent. The agent gathers data from the SNMP MIB, the repository

for information about device parameters and network data. The agent can also respond to manager requests to get or set data.

Note

- PTP over REP or HSR is not supported.

SNMP MIBs Supported with PTP Modes

This section lists the SNMP MIBs supported in different PTP modes.

The following MIBs are supported when the switch is configured with PTP boundary clock mode:

MIB	OID
cPtpClockNodeTable	1.3.6.1.4.1.9.9.760.1.1.3
cPtpClockCurrentDSTable	1.3.6.1.4.1.9.9.760.1.2.1
cPtpClockParentDSTable	1.3.6.1.4.1.9.9.760.1.2.2
cPtpClockDefaultDSTable	1.3.6.1.4.1.9.9.760.1.2.3
cPtpClockTimePropertiesDSTable	1.3.6.1.4.1.9.9.760.1.2.5
cPtpClockPortTable	1.3.6.1.4.1.9.9.760.1.2.7
cPtpClockPortRunningTable	1.3.6.1.4.1.9.9.760.1.2.9

The following MIBs are supported when the switch is configured with PTP transparent clock mode:

MIB	OID
cPtpClockNodeTable	1.3.6.1.4.1.9.9.760.1.1.3
cPtpClockParentDSTable	1.3.6.1.4.1.9.9.760.1.2.2
cPtpClockDefaultDSTable	1.3.6.1.4.1.9.9.760.1.2.3
cPtpClockPortTable	1.3.6.1.4.1.9.9.760.1.2.7
cPtpClockPortRunningTable	1.3.6.1.4.1.9.9.760.1.2.9
cPtpClockSystemTimePropertiesTable	1.3.6.1.4.1.9.9.760.1.2.12

Prerequisites for Configuring SNMP PTP MIBs

Before you configure SNMP PTP MIBs, you should be familiar with the PTP protocol and configurations.

You should also be familiar with the Cisco SNMP Object Navigator, which translates an object identifier (OID) into object name or an object name into OID, enabling you to receive PTP object details. OIDs identify manged objects in an MIB.

Verifying the Configuration

PTP Configuration

You can use the following commands to verify the PTP configuration:

- show ptp clock dataset parent
- show ptp clock dataset current
- show ptp clock dataset time-properties
- show ptp clock dataset default
- show ptp clock running
- show ptp port dataset port
- show ptp lan clock
- show ptp lan port counters messages
- show ptp lan port counters errors
- show ptp lan foreign-master-record
- show ptp lan rogue-master-record
- show ptp lan histogram ?

  delay—Show PTP histogram of mean path delay

  offset—Show PTP histogram of offset

  time-error—Show PTP history of time error (last 15 days)
- show ptp lan history ?

  delay—Show PTP history of mean path delay (last 15 days)

  offset—Show PTP history of offset (last 15 days)

  time-error—Show PTP history of time error (last 15 days)

Default Profile Configuration

The following example shows the Default profile configuration:

```
Default profile MASTER
```

```
Switch# show run | sec ptp
ptp clock boundary domain 0  profile default
 clock-port 1
  transport ipv4 multicast interface Gi1/1
Switch#
Switch#sh ptp clock run

                     PTP Boundary Clock [Domain 0]  [Profile: default]
```

```
          State          Ports          Pkts sent      Pkts rcvd      Redundancy Mode

          FREERUN        1              140            30             Hot standby

                                PORT SUMMARY
                                                                          PTP Master
Name  Tx Mode      Role         Transport    State         Sessions      Port Addr

1     mcast       negotiated   Gi1/1        Master        1             UNKNOWN
Switch#
Switch#sh ptp lan port
  PTP PORT DATASET: GigabitEthernet1/1
    Port identity: clock identity: 0x84:eb:ef:ff:fe:d2:e0:3f
    Port identity: port number: 1
    PTP version: 2
    Port state: MASTER
    Delay request interval(log mean): 0
    Announce receipt time out: 3
    Announce interval(log mean): 1
    Sync interval(log mean): 0
    Delay Mechanism: End to End
    Peer delay request interval(log mean): 0
    Sync fault limit: 500000
    Rogue master block: FALSE
    Ingress phy latency: 590
    Egress phy latency: 0

Switch#
```

Default profile SLAVE

```
Switch#sh run | sec ptp
ptp clock boundary domain 0  profile default
 clock-port 1
  transport ipv4 multicast interface Gi1/1
Switch#
Switch#sh ptp clock run

                     PTP Boundary Clock [Domain 0]  [Profile: default]

          State          Ports          Pkts sent      Pkts rcvd      Redundancy Mode

          PHASE_ALIGNED  1              72             272            Hot standby

                                PORT SUMMARY
                                                                          PTP Master
Name  Tx Mode      Role         Transport    State         Sessions      Port Addr

1     mcast       negotiated   Gi1/1        Slave         1             UNKNOWN
Switch#
Switch#sh ptp lan po
Switch#sh ptp lan port
  PTP PORT DATASET: GigabitEthernet1/1
    Port identity: clock identity: 0x84:eb:ef:ff:fe:d2:e5:3f
    Port identity: port number: 0
    PTP version: 2
    Port state: SLAVE
    Delay request interval(log mean): 0
    Announce receipt time out: 3
    Announce interval(log mean): 1
    Sync interval(log mean): 0
    Delay Mechanism: End to End
    Peer delay request interval(log mean): 0
    Sync fault limit: 500000
```

```
    Rogue master block: FALSE
    Ingress phy latency: 590
    Egress phy latency: 0

Switch#
```

Power Profile Configuration

The following example shows the Power profile configuration:

```
Power profile MASTER
 ____________

Switch#show run | sec ptp
ptp clock boundary domain 0  profile power
 clock-port 1
  transport ethernet multicast interface Gi1/1
Switch#
Switch#
Switch# sh ptp clock running


                      PTP Boundary Clock [Domain 0]  [Profile: power]

         State          Ports          Pkts sent      Pkts rcvd      Redundancy Mode

         FREERUN        1              875328         1578627        Hot standby

                               PORT SUMMARY
                                                                        PTP Master
Name  Tx Mode      Role         Transport    State         Sessions     Port Addr

1     mcast       negotiated   Ethernet     Master        1            UNKNOWN
Switch#
Switch#
Switch#
Switch#
Switch#sh ptp lan port
  PTP PORT DATASET: GigabitEthernet1/1
    Port identity: clock identity: 0x84:eb:ef:ff:fe:d2:e0:3f
    Port identity: port number: 1
    PTP version: 2
    Port state: MASTER
    Delay request interval(log mean): 0
    Announce receipt time out: 3
    Peer mean path delay(ns): 35
    Announce interval(log mean): 0
    Sync interval(log mean): 0
    Delay Mechanism: Peer to Peer
    Peer delay request interval(log mean): 0
    Sync fault limit: 10000
    Rogue master block: FALSE
    Ingress phy latency: 590
    Egress phy latency: 0

Switch#
Switch#
Switch#

Power profile SLAVE
_____________
```

```
Switch#show run | sec ptp
ptp clock boundary domain 0  profile power
 clock-port 1
  transport ethernet multicast interface Gi1/1
Switch#
Switch#
Switch#show ptp clock run


                     PTP Boundary Clock [Domain 0]  [Profile: power]

         State          Ports          Pkts sent      Pkts rcvd      Redundancy Mode

         PHASE_ALIGNED  1              57056          113937         Hot standby

                              PORT SUMMARY
                                                                        PTP Master
Name  Tx Mode      Role          Transport    State          Sessions     Port Addr

1     mcast        negotiated    Ethernet     Slave          1            UNKNOWN
Switch#
Switch#
Switch#
Switch#show ptp lan port
  PTP PORT DATASET: GigabitEthernet1/1
    Port identity: clock identity: 0x84:eb:ef:ff:fe:d2:e5:3f
    Port identity: port number: 0
    PTP version: 2
    Port state: SLAVE
    Delay request interval(log mean): 0
    Announce receipt time out: 3
    Peer mean path delay(ns): 35
    Announce interval(log mean): 0
    Sync interval(log mean): 0
    Delay Mechanism: Peer to Peer
    Peer delay request interval(log mean): 0
    Sync fault limit: 10000
    Rogue master block: FALSE
    Ingress phy latency: 590
    Egress phy latency: 0

Switch#
```

PTP Alarm Configuration

Use the following `show` commands to verify the PTP alarm configuration.

- `show facility-alarm status`

```
Switch#show facility-alarm status
Source                  Severity   Description                          Relay    Time
GigabitEthernet1/1    MINOR      5 PTP SLAVE port state changed        MIN      Jan 01
 1970 21:17:59
```

- `show ptp clock running`

```
Switch#show ptp clock running
 PTP Boundary Clock [Domain 10]  [Profile: power]
         State          Ports          Pkts sent      Pkts rcvd      Redundancy Mode
         PHASE_ALIGNED  2              1806           2615           Hot standby
                              PORT SUMMARY
                                                                        PTP Master
Name  Tx Mode      Role          Transport    State          Sessions     Port Addr
21    mcast        negotiated    Ethernet     Slave          1            UNKNOWN
```

```
Switch#
```

- `show ptp clock running`

```
Switch#show ptp clock running
 PTP Boundary Clock [Domain 10]  [Profile: power]
         State          Ports          Pkts sent      Pkts rcvd      Redundancy Mode
         PHASE_ALIGNED  2              1806           2615           Hot standby
                                PORT SUMMARY
                                                                          PTP Master
Name  Tx Mode      Role         Transport    State          Sessions     Port Addr
21    mcast       negotiated   Ethernet     Slave          1            UNKNOWN
```

Troubleshooting PTP

This section contains instructions for troubleshooting PTP by checking if the Transparent Clock is receiving messages from the Grandmaster Clock, verifying packet message and error counters, and running debug commands.

Verify that the Transparent Clock is Syntonized

You might want to verify that the Transparent Clock is syntonized to the Grand Master Clock—that is, that the Transparent Clock is logged to the Grand Master Clock. You might want to verify syntonization because the `show ptp clock running` command does not appy to the Transparent Clock. Subordinate clocks in the PTP network do not synchronize with the Grand Master Clock if the Transparent Clock is not syntonized.

Procedure

Verify that the Transparent Clock is syntonized.

Example:

```
switch(config-ptp-port)# sh ptp clock dataset time-properties
Clock Syntonized: TRUE
```

The command output is `TRUE` if the Transparent Clock is syntonized and `FALSE` if it is not. You also can check counters to see if PTP messages are being received.

Verify PTP Messages

You can verify whether messages are being received from the Grandmaster Clock.

Procedure

Verify PTP LAN port packet message.

Example:

```
switch# show ptp lan port counters messages

GigabitEthernet1/1

    Transmit                          Receive
    250    Announce                   0    Announce
    248    Sync                       0    Sync
    248    Follow_Up                  0    Follow_Up
     0     Delay_Req                  0    Delay_Req
     0     Delay_Resp                 0    Delay_Resp
    251    Pdelay_Req                251   Pdelay_Req
    251    Pdelay_Resp               251   Pdelay_Resp
    251    Pdelay_Resp_Follow_Up     251   Pdelay_Resp_Follow_Up
     0     Signaling                  0    Signaling
     0     Management                 0    Management
```

The preceding example shows that all the packets are being received.

The output of the command would vary, depending on which packets are not received. The following example shows output if follow-ups are not received.

```
GigabitEthernet1/1

Transmit Receive
0 Announce 1359 Announce
0 Sync 1359 Sync
0 Follow_Up 0 Follow_Up <<<
0 Delay_Req 0 Delay_Req
0 Delay_Resp 0 Delay_Resp
1362 Pdelay_Req 1359 Pdelay_Req
1359 Pdelay_Resp 1360 Pdelay_Resp
1359 Pdelay_Resp_Follow_Up 1360 Pdelay_Resp_Follow_Up
0 Signaling 0 Signaling
0 Management 0 Management
```

Note
You can use the following command to reset the counters: **clear ptp all all-clocks**

Verify PTP Error Counters

You can verify whether the error counters are continously incrementing, indicating that messages from the Grandmaster Clock aren't being received.

Procedure

Verify PTP LAN port

Example:

```
switch# show ptp lan port counters errors

GigabitEthernet1/1

     0   Sanity check failed      0   Blocked port
     0   Timestamp get failed     0   ParentId invalid
```

```
0   Vlan mismatch            0   Gmcld invalid
0   Domain mismatch          0   SequenceId invalid
0   Sync fault               0   Unmatched Follow_Up
0   Duplicate Sync           0   Unmatched Delay_Resp
0   Duplicate Announce       0   Unmatched Pdelay_Resp
0   Send error               0   Unmatched Pdelay_Resp_Follow_Up
0   Misc error               0   Rogue master Sync
0   Rogue master Follow-Up   0   Rogue master Announce
```

The preceding example shows that no error counters are being incremented.

The following example shows how errors increment when the VLAN in the ingress PTP message is different from the PTP VLAN used on the port.

```
switch# sh ptp lan port counters errors | beg 1/1
GigabitEthernet1/1

0 Sanity check failed 0 Blocked port
0 Timestamp get failed 0 ParentId invalid
1482 Vlan mismatch 0 GmcId invalid
0 Domain mismatch 0 SequenceId invalid
0 Sync fault 0 Unmatched Follow_Up
0 Duplicate Sync 0 Unmatched Delay_Resp
0 Duplicate Announce 0 Unmatched Pdelay_Resp
0 Send error 0 Unmatched Pdelay_Resp_Follow_Up
0 Misc error 0 Rogue master Sync
0 Rogue master Follow_Up 0 Ro
```

Note
You can use the following command to reset the counters: **clear ptp all all-clocks**

Debugging Commands

The debugging feature collects logs that can be analyzed to resolve any issues on the switch. You can enable debugging on the switch, which logs debugging lists to a file on the switch or to a boot device.

Note

- We recommend that you save the debugging information to a boot device rather than to an internal file.

  Make sure that you have enough space on the boot device for the debugging logs.
- Enable debugging only when you are troubleshooting and disable debugging when you finish. Disabling debugging when not troubleshooting reduces CPU overhead.

Enabling Debugging

Enter both of the following commands to enable debugging on the switch:

- `switch# set platform software trace timingd switch active R0 iot-ptp debug`
- `switch# set platform software trace timingd switch active R0 timingd debug`

Note When you use the preceding commands, debugging information is not printed on the screen and will be logged to an internal file. You cannot access the file directly, but you can store the debugging information to a boot device, which you can access.

Storing Debugging Information on a Boot Device

Use the following command to store the debugging information in the internal file to a boot device:

Note You can give the debug file any name you choose. The following example uses `timing-logs` as the filename.

```
Switch# show log process timingd internal to-file bootflash:timing-logs
```

When you use the preceding command, the debugging information is printed on the screen in addition to being saved to the boot device.

Checking Debugging

Enter both of the following commands to see if debugging information is being collected:

```
switch#sh platform software trace level timingd switch active R0  | inc iot-ptp
iot-ptp                         Debug

switch#sh platform software trace level timingd switch active R0  | inc timingd
timingd                         Debug
```

Disabling Debugging

Enter both of the following commands to disable debugging on the switch:

- `switch# set platform software trace timingd switch active R0 iot-ptp notice`
- `switch# set platform software trace timingd switch active R0 timingd notice`

CHAPTER 170

NTP Timing Based on PTP Clock

PTP as a Reference Clock for NTP

You can configure Precision Timing Protocol (PTP) time as the reference clock for Network Time Protocol (NTP).

PTP time acts as a stratum 0 source, and the Cisco IOS NTP server acts as a stratum 1 device. The server then provides clock information to its NTP clients (strata 2 and 3).

A Network Essentials or Network Advantage license is required.

Enabling PTP as a Reference Clock for NTP

The PTP reference clock feature is disabled by default. You enable it by entering a CLI command.

Before you begin

- Configure PTP and ensure that it is in slave mode.

  See the chapter *Precision Time Protocol* in this guide for configuration instructions.

Procedure

To enable PTP as a reference clock for NTP, enter the following command:**ntp refclock ptp**.

You disable the PTP reference clock feature by entering the following command: **no ntp refclock ptp**.

What to do next

Validate the PTP reference clock feature on the switch. See *Validate the PTP Reference Clock*.

Validate the PTP Reference Clock

After you enable PTP as the reference clock for NTP, you can enter CLI commands to validate the configuration.

Procedure

Step 1 Check that the PTP reference clock configuration is correct and that the feature is running.

Example:

```
#show run | sec ptp|ntp
ntp refclock ptp
ptp clock boundary domain 0  profile power
 clock-port 1
  transport ethernet multicast interface Gi1/1
```

Step 2 Check that PTP is in slave mode; that is PTP is in phase aligned state, which means it is locked to a master clock.

Example:

```
#sh ptp clock running
                      PTP Boundary Clock [Domain 0]  [Profile: power]
         State          Ports          Pkts sent      Pkts rcvd      Redundancy Mode
         PHASE_ALIGNED  1              629978         633            Hot standby
                                PORT SUMMARY
                                                                          PTP Master
Name  Tx Mode      Role         Transport    State         Sessions     Port Addr
1     mcast        negotiated   Ethernet     Slave         1            UNKNOWN
```

Step 3 Check that NTP is using PTP. as its reference clock.

Example:

```
#show ntp status
Clock is synchronized, stratum 1, reference is .PTP.
nominal freq is 250.0000 Hz, actual freq is 249.9998 Hz, precision is 2**10
ntp uptime is 28233900 (1/100 of seconds), resolution is 4016
reference time is E6161FA8.FFBE7988 (08:26:16.999 UTC Fri Apr 29 2022)
clock offset is 0.9998 msec, root delay is 0.00 msec
root dispersion is 3940.49 msec, peer dispersion is 3938.47 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000856 s/s
system poll interval is 64, last update was 4 sec ago.
```

Troubleshooting PTP as an NTP Reference Clock

Checking PTP-NTP Synchronization

You can check the time on the PTP and NTP cocks to ensure that they are synchronized, as shown in the following example.

```
#show ptp lan clock | inc time
 Local clock time: 2022-4-29 8:48:39 UTC
#
#show clock detail
08:48:39.278 UTC Fri Apr 29 2022
Time source is NTP
#
```

Troubleshooting Commands

Table 178:

Command	Description
`ntp logging`	Enables syslogs from NTP.
`debug ntp all`	Provides the complete debugging logs for NTP processes.
`debug platform software pd-ptp all`	Provides debugging logs on the switch relating to PTP as a reference clock.
`show ntp status`	Shows detailed NTP status, including whether NTP is using PTP as its reference clock.
`show ntp association detail`	Shows detailed information about NTP peering.
`show ptp clock running`	Check that PTP is in slave mode; that is PTP is in phase aligned state, which means it is locked to a master clock.

Viewing Peering Details

The command output shows detailed information about NTP peering. You can use the command to check the amount of time the platform takes to switch to the next available timing source after the initial timing source goes down. In the following example, NTP waits 8x256 seconds to switch over to the next source .

```
#show ntp assoc deta
127.127.6.1 configured, ipv4, our_master, sane, valid, stratum 0
ref ID .PTP., time E61622E9.00000000 (08:40:09.000 UTC Fri Apr 29 2022)
our mode active, peer mode passive, our poll intvl 256, peer poll intvl 1024
root delay 0.00 msec, root disp 0.00, reach 377, sync dist 4.62
delay 0.00 msec, offset 0.9998 msec, dispersion 2.81, jitter 0.97 msec
precision 2**10, version 4
assoc id 63756, assoc name 127.127.6.1
assoc in packets 11, assoc out packets 17652, assoc error packets 0
org time E61622E8.FFBE7988 (08:40:08.999 UTC Fri Apr 29 2022)
rec time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
```

```
xmt time E61622E8.FFBE7988 (08:40:08.999 UTC Fri Apr 29 2022)
filtdelay =     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filtoffset =    0.99    1.99    0.99    0.99    0.99    0.99    1.99    0.99
filterror =     0.97    2.89    4.81    6.73    8.65   10.57   11.53   12.49
minpoll = 4, maxpoll = 10
```

CHAPTER 171

MODBUS

MODBUS Protocol

Modicon Communication Bus (MODBUS) is an application layer protocol for client-server communication between a switch (server) and a device in the network running MODBUS client software (client). You can use MODBUS over a serial line to connect a computer to a remote terminal unit (RTU) in supervisory control and data acquisition (SCADA) systems.

MODBUS also runs on Ethernet TCP/IP networks. Use MODBUS TCP over an Ethernet network when connecting the switch to devices such as intelligent electronic devices (IEDs), distributed controllers, substation routers, IP phones, Wireless Access Points, and other network devices such as redundant substation switches.

The client can be an IED or a human machine interface (HMI) application that remotely configures and manages devices running MODBUS TCP. The switch functions as the server.

The switch encapsulates a request or response message in a MODBUS TCP application data unit (ADU). A client sends a message to a TCP port on the switch.

MODBUS TCP Registers

This document lists the read-only registers for switches. MODBUS clients use them to communicate with a MODBUS server (the switch). There are no writable registers.

System Information Registers

Address spaces 0x0800 through 0x0FFF are system information read-only registers. These 2048 registers are accessible by MODBUS function code 0x03 `Read Multiple Registers`.

Table 179: System Information Registers

Address	Number of Registers	Description	R/W	Format	Value
0x0800	64	Product ID	R	Text	
0x0840	64	Software image name	R	Text	
0x0880	64	Software image version	R	Text	
0x08C0	64	Host Name	R	Text	
0x0900	1	Number of Gigabit Ethernet ports	R	Uint 16	
0x0901	1	Number of 10 Gigabit Ethernet ports	R	Uint 16	
0x0902	1	Number of mgig Ethernet ports	R	Uint 16	
0x0903	1	Number of power supplies	R	Uint 16	
0x0904	1	Power supply 1 status	R	Uint 16	0 – Not present 1 – Present
0x0905	1	Power supply 2 status	R	Uint 16	0 – Not present 1 – Present
0x0906	1	System or sensor ID 0 temperature	R	Uint 16	
0x0907	1	Alarm 1 – Description	R	Uint 16	
0x0947	2	Alarm 2 Description	R	Uint 16	
0x0987	1	Number of Alarms	R	Uint 16	
0x0988	1	Alarm 1 Status	R	Uint 16	0 – Alarm off 1 – Alarm on
0x0989	2	Alarm 2 Status	R	Uint 16	0 – Alarm off 1 – Alarm on

Address	Number of Registers	Description	R/W	Format	Value
0x098A	64	Alarm 3 Description	R	Text	
0x09CA	64	Alarm 4 Description	R	Text	
0x0A0A	1	Alarm 3 Status	R	Uint16	0 – Alarm off 1 – Alarm on
0x0A0B	1	Alarm 4 Status	R	Uint16	0 – Alarm off 1 – Alarm on

Port Information Registers

Address spaces 0x1000 through 0x3FFF are port information read-only registers. These 12,000 registers are accessible by MODBUS function code 0x03 `Read Multiple Registers`. They are updated every time upon receiving a read request to the register(s).

The following table shows the memory map for all interface registers, with 64-bit counters (Address space 0x1000 through 0x2FFF, 8,000 registers).

Table 180: 11 Port Registers

Address	Number of Registers	Description	R/W	Format
1000	64	Port 1 Name	R	Text
1040	64	Port 2 Name	R	Text
1080	64	Port 3 Name	R	Text
10C0	64	Port 4 Name	R	Text
1100	64	Port 5 Name	R	Text
1140	64	Port 6 Name	R	Text
1180	64	Port 7 Name	R	Text
11C0	64	Port 8 Name	R	Text
1200	64	Port 9 Name	R	Text
1240	64	Port 10 Name	R	Text
1280	64	Port 11 Name	R	Text
12C0	1	Port 1 State	R	Uint16
12C1	1	Port 2 State	R	Uint16
12C2	1	Port 3 State	R	Uint16
12C3	1	Port 4 State	R	Uint16

Address	Number of Registers	Description	R/W	Format
12C4	1	Port 5 State	R	Uint16
12C5	1	Port 6 State	R	Uint16
12C6	1	Port 7 State	R	Uint16
12C7	1	Port 8 State	R	Uint16
12C8	1	Port 9 State	R	Uint16
12C9	1	Port 10 State	R	Uint16
12CA	1	Port 11 State	R	Uint16
12CB	4	Port 1 Statistics – Number of packets received	R	Uint64
12CF	4	Port 2 Statistics – Number of packets received	R	Uint64
12D3	4	Port 3 Statistics – Number of packets received	R	Uint64
12D7	4	Port 4 Statistics – Number of packets received	R	Uint64
12DB	4	Port 5 Statistics – Number of packets received	R	Uint64
12DF	4	Port 6 Statistics – Number of packets received	R	Uint64
12E3	4	Port 7 Statistics – Number of packets received	R	Uint64
12E7	4	Port 8 Statistics – Number of packets received	R	Uint64
12EB	4	Port 9 Statistics – Number of packets received	R	Uint64
12EF	4	Port 10 Statistics – Number of packets received	R	Uint64

Address	Number of Registers	Description	R/W	Format
12F3	4	Port 11 Statistics – Number of packets received	R	Uint64
12F7	4	Port 1 Statistics – Number of packets sent	R	Uint64
12FB	4	Port 2 Statistics – Number of packets sent	R	Uint64
12FF	4	Port 3 Statistics – Number of packets sent	R	Uint64
1303	4	Port 4 Statistics – Number of packets sent	R	Uint64
1307	4	Port 5 Statistics – Number of packets sent	R	Uint64
130B	4	Port 6 Statistics – Number of packets sent	R	Uint64
130F	4	Port 7 Statistics – Number of packets sent	R	Uint64
1313	4	Port 8 Statistics – Number of packets sent	R	Uint64
1317	4	Port 9 Statistics – Number of packets sent	R	Uint64
131B	4	Port 10 Statistics – Number of packets sent	R	Uint64
131F	4	Port 11 Statistics – Number of packets sent	R	Uint64
1323	4	Port 1 Statistics – Number of packets received	R	Uint64

Address	Number of Registers	Description	R/W	Format
1327	4	Port 2 Statistics – Number of packets received	R	Uint64
132B	4	Port 3 Statistics – Number of packets received	R	Uint64
132F	4	Port 4 Statistics – Number of packets received	R	Uint64
1333	4	Port 5 Statistics – Number of packets received	R	Uint64
1337	4	Port 6 Statistics – Number of packets received	R	Uint64
133B	4	Port 7 Statistics – Number of packets received	R	Uint64
133F	4	Port 8 Statistics – Number of packets received	R	Uint64
1343	4	Port 9 Statistics – Number of packets received	R	Uint64
1347	4	Port 10 Statistics – Number of packets received	R	Uint64
134B	4	Port 11 Statistics – Number of packets received	R	Uint64
134F	1	Port 1 Statistics – Number of bytes sent	R	Uint64
1353	1	Port 2 Statistics – Number of bytes sent	R	Uint64
1357	1	Port 3 Statistics – Number of bytes sent	R	Uint64

Address	Number of Registers	Description	R/W	Format
135B	1	Port 4 Statistics – Number of bytes sent	R	Uint64
135F	1	Port 5 Statistics – Number of bytes sent	R	Uint64
1363	1	Port 6 Statistics – Number of bytes sent	R	Uint64
1367	1	Port 7 Statistics – Number of bytes sent	R	Uint64
136B	1	Port 8 Statistics – Number of bytes sent	R	Uint64
136F	1	Port 9 Statistics – Number of bytes sent	R	Uint64
1373	1	Port 10 Statistics – Number of bytes sent	R	Uint64
1377	1	Port 11 Statistics – Number of bytes sent	R	Uint64

Table 181: Values for Getting 15 Port Registers

Address	Number of Registers	Description	R/W	Format
1000	64	Port 1 Name	R	Text
1040	64	Port 2 Name	R	Text
1080	64	Port 3 Name	R	Text
10C0	64	Port 4 Name	R	Text
1100	64	Port 5 Name	R	Text
1140	64	Port 6 Name	R	Text
1180	64	Port 7 Name	R	Text
11C0	64	Port 8 Name	R	Text

Address	Number of Registers	Description	R/W	Format
1200	64	Port 9 Name	R	Text
1240	64	Port 10 Name	R	Text
1280	64	Port 11 Name	R	Text
12C0	64	Port 12 Name	R	Text
1300	64	Port 13 Name	R	Text
1340	64	Port 14 Name	R	Text
1380	64	Port 15 Name	R	Text
13C0	1	Port 1 Status	R	Text
13C1	1	Port 2 Status	R	Text
13C2	1	Port 3 Status	R	Text
13C3	1	Port 4 Status	R	Text
13C4	1	Port 5 Status	R	Uint16
13C5	1	Port 6 Status	R	Uint16
13C6	1	Port 7 Status	R	Uint16
13C7	1	Port 8 Status	R	Uint16
13C8	1	Port 9 Status	R	Uint16
13C9	1	Port 10 Status	R	Uint16
13CA	1	Port 11 Status	R	Uint16
13CB	1	Port 12 Status	R	Uint16
13CC	1	Port 13 Status	R	Uint16
13CD	1	Port 14 State	R	Uint16
13CE	1	Port 15 State	R	Uint16
13CF	4	Port 1 Statistics – Number of packets received	R	Uint16
13D3	4	Port 2 Statistics – Number of packets received	R	Uint16

Address	Number of Registers	Description	R/W	Format
13D7	4	Port 3 Statistics – Number of packets received	R	Uint16
13DB	4	Port 4 Statistics – Number of packets received	R	Uint16
13DF	4	Port 5 Statistics – Number of packets received	R	Uint16
13E 3	4	Port 6 Statistics – Number of packets received	R	Uint16
13E 7	4	Port 7 Statistics – Number of packets received	R	Uint16
13E B	4	Port 8 Statistics – Number of packets received	R	Uint16
13E F	4	Port 9 Statistics – Number of packets received	R	Uint64
13F3	4	Port 10 Statistics – Number of packets received	R	Uint64
13F7	4	Port 11 Statistics – Number of packets received	R	Uint64
13FB	4	Port 12 Statistics – Number of packets received	R	Uint64
13FF	4	Port 13 Statistics – Number of packets received	R	Uint64
1403	4	Port 14 Statistics – Number of packets received	R	Uint64
1407	4	Port 15 Statistics – Number of packets received	R	Uint64

Address	Number of Registers	Description	R/W	Format
140B	4	Port 1 Statistics – Number of packets sent	R	Uint64
140F	4	Port 2 Statistics – Number of packets sent	R	Uint64
1413	4	Port 3 Statistics – Number of packets sent	R	Uint64
1417	4	Port 4 Statistics – Number of packets sent	R	Uint64
141B	4	Port 5 Statistics – Number of packets sent	R	Uint64
141F	4	Port 6 Statistics – Number of packets sent	R	Uint64
1423	4	Port 7 Statistics – Number of packets sent	R	Uint64
1427	4	Port 8 Statistics – Number of packets sent	R	Uint64
142B	4	Port 9 Statistics – Number of packets sent	R	Uint64
142F	4	Port 10 Statistics – Number of packets sent	R	Uint64
1433	4	Port 11 Statistics – Number of packets sent	R	Uint64
1437	4	Port 12 Statistics – Number of packets sent	R	Uint64
143B	4	Port 13 Statistics – Number of packets sent	R	Uint64

Address	Number of Registers	Description	R/W	Format
143F	4	Port 14 Statistics – Number of packets sent	R	Uint64
1443	4	Port 15 Statistics – Number of packets sent	R	Uint64
1447	4	Port 1 Statistics – Number of bytes received	R	Uint64
144B	4	Port 2 Statistics – Number of bytes received	R	Uint64
144F	4	Port 3 Statistics – Number of bytes received	R	Uint64
1453	4	Port 4 Statistics – Number of bytes received	R	Uint64
1457	4	Port 5 Statistics – Number of bytes received	R	Uint64
145B	4	Port 6 Statistics – Number of bytes received	R	Uint64
145F	4	Port 7 Statistics – Number of bytes received	R	Uint64
1463	4	Port 8 Statistics – Number of bytes received	R	Uint64
1467	4	Port 9 Statistics – Number of bytes received	R	Uint64
146B	4	Port 10 Statistics – Number of bytes received	R	Uint64
146F	4	Port 11 Statistics – Number of bytes received	R	Uint64

Address	Number of Registers	Description	R/W	Format
1473	4	Port 12 Statistics – Number of bytes received	R	Uint64
1477	4	Port 13 Statistics – Number of bytes received	R	Uint64
147B	4	Port 14 Statistics – Number of bytes received	R	Uint64
147F	4	Port 15 Statistics – Number of bytes received	R	Uint64
1483	4	Port 1 Statistics – Number of bytes sent	R	Uint64
1487	4	Port 2 Statistics – Number of bytes sent	R	Uint64
148B	4	Port 3 Statistics – Number of bytes sent	R	Uint64
148F	4	Port 4 Statistics – Number of bytes sent	R	Uint64
1493	4	Port 5 Statistics – Number of bytes sent	R	Uint64
1497	4	Port 6 Statistics – Number of bytes sent	R	Uint64
149B	4	Port 7 Statistics – Number of bytes sent	R	Uint64
149F	4	Port 8 Statistics – Number of bytes sent	R	Uint64
14A3	4	Port 9 Statistics – Number of bytes sent	R	Uint64

Address	Number of Registers	Description	R/W	Format
14A7	4	Port 10 Statistics – Number of bytes sent	R	Uint64
14AB	4	Port 11 Statistics – Number of bytes sent	R	Uint64
14AF	4	Port 12 Statistics – Number of bytes sent	R	Uint64
14B3	4	Port 13 Statistics – Number of bytes sent	R	Uint64
14B7	4	Port 14 Statistics – Number of bytes sent	R	Uint64
14BB	4	Port 15 Statistics – Number of bytes sent	R	Uint64

Table 182: Values for Getting 19 Port Information

Address	Number of Registers	Description	R/W	Format
1000	64	Port 1 Name	R	Text
1040	64	Port 2 Name	R	Text
1080	64	Port 3 Name	R	Text
10C0	64	Port 4 Name	R	Text
1100	64	Port 5 Name	R	Text
1140	64	Port 6 Name	R	Text
1180	64	Port 7 Name	R	Text
11C0	64	Port 8 Name	R	Text
1200	64	Port 9 Name	R	Text
1240	64	Port 10 Name	R	Text
1280	64	Port 11 Name	R	Text
12C0	64	Port 12 Name	R	Text
1300	64	Port 13 Name	R	Text

Address	Number of Registers	Description	R/W	Format
1340	64	Port 14 Name	R	Text
1380	64	Port 15 Name	R	Text
13C0	64	Port 16 Name	R	Text
1400	64	Port 17 Name	R	Text
1440	64	Port 18 Name	R	Text
1480	64	Port 19 Name	R	Text
14C0	1	Port 1 State	R	Uint16
14C1	1	Port 2 State	R	Uint16
14C2	1	Port 3 State	R	Uint16
14C3	1	Port 4 State	R	Uint16
14C4	1	Port 5 State	R	Uint16
14C5	1	Port 6 State	R	Uint16
14C6	1	Port 7 State	R	Uint16
14C7	1	Port 8 State	R	Uint16
14C8	1	Port 9 State	R	Uint16
14C9	1	Port 10 State	R	Uint16
14CA	1	Port 11 State	R	Uint16
14CB	1	Port 12 State	R	Uint16
14CC	1	Port 13 State	R	Uint16
14CD	1	Port 14 State	R	Uint16
14CE	1	Port 15 State	R	Uint16
14CF	1	Port 16 State	R	Uint16
14D0	1	Port 17 State	R	Uint16
14D1	1	Port 18 State	R	Uint16
14D2	1	Port 19 State	R	Uint16
14D3	4	Port 1 Statistics – Number of packets received	R	Uint64

Address	Number of Registers	Description	R/W	Format
14D7	4	Port 2 Statistics – Number of packets received	R	Uint64
14DB	4	Port 3 Statistics – Number of packets received	R	Uint64
14DF	4	Port 4 Statistics – Number of packets received	R	Uint64
14E3	4	Port 5 Statistics – Number of packets received	R	Uint64
14E7	4	Port 6 Statistics – Number of packets received	R	Uint64
14EB	4	Port 7 Statistics – Number of packets received	R	Uint64
14EF	4	Port 8 Statistics – Number of packets received	R	Uint64
14F3	4	Port 9 Statistics – Number of packets received	R	Uint64
14F7	4	Port 10 Statistics – Number of packets received	R	Uint64
14FB	4	Port 11 Statistics – Number of packets received	R	Uint64
14FF	4	Port 12 Statistics – Number of packets received	R	Uint64
1503	4	Port 13 Statistics – Number of packets received	R	Uint64
1507	4	Port 14 Statistics – Number of packets received	R	Uint64

Address	Number of Registers	Description	R/W	Format
150B	4	Port 15 Statistics – Number of packets received	R	Uint64
150F	4	Port 16 Statistics – Number of packets received	R	Uint64
1513	4	Port 17 Statistics – Number of packets received	R	Uint64
1517	4	Port 18 Statistics – Number of packets received	R	Uint64
151B	4	Port 19 Statistics – Number of packets received	R	Uint64
151F	4	Port 1 Statistics – Number of packets sent	R	Uint64
1523	4	Port 2 Statistics – Number of packets sent	R	Uint64
1527	4	Port 3 Statistics – Number of packets sent	R	Uint64
152B	4	Port 4 Statistics – Number of packets sent	R	Uint64
152F	4	Port 5 Statistics – Number of packets sent	R	Uint64
1533	4	Port 6 Statistics – Number of packets sent	R	Uint64
1537	4	Port 7 Statistics – Number of packets sent	R	Uint64
153B	4	Port 8 Statistics – Number of packets sent	R	Uint64

Address	Number of Registers	Description	R/W	Format
153F	4	Port 9 Statistics – Number of packets sent	R	Uint64
1543	4	Port 10 Statistics – Number of packets sent	R	Uint64
1547	4	Port 11 Statistics – Number of packets sent	R	Uint64
154B	4	Port 12 Statistics – Number of packets sent	R	Uint64
154F	4	Port 13 Statistics – Number of packets sent	R	Uint64
1553	4	Port 14 Statistics – Number of packets sent	R	Uint64
1557	4	Port 15 Statistics – Number of packets sent	R	Uint64
155B	4	Port 16 Statistics – Number of packets sent	R	Uint64
155F	4	Port 17 Statistics – Number of packets sent	R	Uint64
1563	4	Port 18 Statistics – Number of packets sent	R	Uint64
1567	4	Port 19 Statistics – Number of packets sent	R	Uint64
156B	4	Port 1 Statistics – Number of bytes received	R	Uint64
156F	4	Port 2 Statistics – Number of bytes received	R	Uint64

Address	Number of Registers	Description	R/W	Format
1573	4	Port 3 Statistics – Number of bytes received	R	Uint64
1577	4	Port 4 Statistics – Number of bytes received	R	Uint64
157B	4	Port 5 Statistics – Number of bytes received	R	Uint64
157F	4	Port 6 Statistics – Number of bytes received	R	Uint64
1583	4	Port 7 Statistics – Number of bytes received	R	Uint64
1587	4	Port 8 Statistics – Number of bytes received	R	Uint64
158B	4	Port 9 Statistics – Number of bytes received	R	Uint64
158F	4	Port 10 Statistics – Number of bytes received	R	Uint64
1593	4	Port 11 Statistics – Number of bytes received	R	Uint64
1597	4	Port 12 Statistics – Number of bytes received	R	Uint64
159B	4	Port 13 Statistics – Number of bytes received	R	Uint64
159F	4	Port 14 Statistics – Number of bytes received	R	Uint64
15A3	4	Port 15 Statistics – Number of bytes received	R	Uint64

Address	Number of Registers	Description	R/W	Format
15A7	4	Port 16 Statistics – Number of bytes received	R	Uint64
15AB	4	Port 17 Statistics – Number of bytes received	R	Uint64
15AF	4	Port 18 Statistics – Number of bytes received	R	Uint64
15B3	4	Port 19 Statistics – Number of bytes received	R	Uint64
15B7	4	Port 1 Statistics – Number of bytes sent	R	Uint64
15BB	4	Port 2 Statistics – Number of bytes sent	R	Uint64
15BF	4	Port 3 Statistics – Number of bytes sent	R	Uint64
15C3	4	Port 4 Statistics – Number of bytes sent	R	Uint64
15C7	4	Port 5 Statistics – Number of bytes sent	R	Uint64
15CB	4	Port 6 Statistics – Number of bytes sent	R	Uint64
15CF	4	Port 7 Statistics – Number of bytes sent	R	Uint64
15D3	4	Port 8 Statistics – Number of bytes sent	R	Uint64
15D7	4	Port 9 Statistics – Number of bytes sent	R	Uint64

Address	Number of Registers	Description	R/W	Format
15DB	4	Port 10 Statistics – Number of bytes sent	R	Uint64
15DF	4	Port 11 Statistics – Number of bytes sent	R	Uint64
15E3	4	Port 12 Statistics – Number of bytes sent	R	Uint64
15E7	4	Port 13 Statistics – Number of bytes sent	R	Uint64
15EB	4	Port 14 Statistics – Number of bytes sent	R	Uint64
15EF	4	Port 15 Statistics – Number of bytes sent	R	Uint64
15F3	4	Port 16 Statistics – Number of bytes sent	R	Uint64
15F7	4	Port 17 Statistics – Number of bytes sent	R	Uint64
15FB	4	Port 18 Statistics – Number of bytes sent	R	Uint64

Table 183: Values for Getting Port 27 Information

Address	Number of Registers	Description	R/W	Format
1000	64	Port 1 Name	R	Text
1040	64	Port 2 Name	R	Text
1080	64	Port 3 Name	R	Text
10C0	64	Port 4 Name	R	Text
1100	64	Port 5 Name	R	Text
1140	64	Port 6 Name	R	Text

Address	Number of Registers	Description	R/W	Format
1180	64	Port 7 Name	R	Text
11C0	64	Port 8 Name	R	Text
1200	64	Port 9 Name	R	Text
1240	64	Port 10 Name	R	Text
1280	64	Port 11 Name	R	Text
12C0	64	Port 12 Name	R	Text
1300	64	Port 13 Name	R	Text
1340	64	Port 14 Name	R	Text
1380	64	Port 15 Name	R	Text
13C0	64	Port 16 Name	R	Text
1400	64	Port 17 Name	R	Text
1440	64	Port 18 Name	R	Text
1480	64	Port 19 Name	R	Text
14C0	64	Port 20 Name	R	Text
1500	64	Port 21 Name	R	Text
1540	64	Port 22 Name	R	Text
1580	64	Port 23 Name	R	Text
15C0	64	Port 24 Name	R	Text
1600	64	Port 25 Name	R	Text
1640	64	Port 26 Name	R	Text
1680	64	Port 27 Name	R	Text
16C0	1	Port 1 State	R	Uint16
16C1	1	Port 2 State	R	Uint16
16C2	1	Port 3 State	R	Uint16
16C3	1	Port 4 State	R	Uint16
16C4	1	Port 5 State	R	Uint16
16C5	1	Port 6 State	R	Uint16
16C6	1	Port 7 State	R	Uint16

Address	Number of Registers	Description	R/W	Format
16C7	1	Port 8 State	R	Uint16
16C8	1	Port 9 State	R	Uint16
16C9	1	Port 10 State	R	Uint16
16CA	1	Port 11 State	R	Uint16
16CB	1	Port 12 State	R	Uint16
16CC	1	Port 13 State	R	Uint16
16CD	1	Port 14 State	R	Uint16
16CE	1	Port 15 State	R	Uint16
16CF	1	Port 16 State	R	Uint16
16D0	1	Port 17 State	R	Uint16
16D1	1	Port 18 State	R	Uint16
16D2	1	Port 19 State	R	Uint16
16D3	1	Port 20 State	R	Uint16
16D4	1	Port 21 State	R	Uint16
16D5	1	Port 22 State	R	Uint16
16D6	1	Port 23 State	R	Uint16
16D7	1	Port 24 State	R	Uint16
16D8	1	Port 25 State	R	Uint16
16D9	1	Port 26 State	R	Uint16
16DA	1	Port 27 State	R	Uint16
16DB	4	Port 1 Statistics – Number of packets received	R	Uint64
16DF	4	Port 2 Statistics – Number of packets received	R	Uint64
16E3	4	Port 3 Statistics – Number of packets received	R	Uint64

Address	Number of Registers	Description	R/W	Format
16E7	4	Port 4 Statistics – Number of packets received	R	Uint64
16EB	4	Port 5 Statistics – Number of packets received	R	Uint64
16EF	4	Port 6 Statistics – Number of packets received	R	Uint64
16F3	4	Port 7 Statistics – Number of packets received	R	Uint64
16F7	4	Port 8 Statistics – Number of packets received	R	Uint64
16FB	4	Port 9 Statistics – Number of packets received	R	Uint64
16FF	4	Port 10 Statistics – Number of packets received	R	Uint64
1703	4	Port 11 Statistics – Number of packets received	R	Uint64
1707	4	Port 12 Statistics – Number of packets received	R	Uint64
170B	4	Port 13 Statistics – Number of packets received	R	Uint64
170F	4	Port 14 Statistics – Number of packets received	R	Uint64
1713	4	Port 15 Statistics – Number of packets received	R	Uint64
1717	4	Port 16 Statistics – Number of packets received	R	Uint64

Address	Number of Registers	Description	R/W	Format
171B	4	Port 17 Statistics – Number of packets received	R	Uint64
171F	4	Port 18 Statistics – Number of packets received	R	Uint64
1723	4	Port 19 Statistics – Number of packets received	R	Uint64
1727	4	Port 20 Statistics – Number of packets received	R	Uint64
172B	4	Port 21 Statistics – Number of packets received	R	Uint64
172F	4	Port 22 Statistics – Number of packets received	R	Uint64
1733	4	Port 23 Statistics – Number of packets received	R	Uint64
1737	4	Port 24 Statistics – Number of packets received	R	Uint64
173B	4	Port 25 Statistics – Number of packets received	R	Uint64
173F	4	Port 26 Statistics – Number of packets received	R	Uint64
1743	4	Port 27 Statistics – Number of packets received	R	Uint64
1747	4	Port 1 Statistics – Number of packets sent	R	Uint64
174B	4	Port 2 Statistics – Number of packets sent	R	Uint64

Address	Number of Registers	Description	R/W	Format
174F	4	Port 3 Statistics – Number of packets sent	R	Uint64
1753	4	Port 4 Statistics – Number of packets sent	R	Uint64
1757	4	Port 5 Statistics – Number of packets sent	R	Uint64
175B	4	Port 6 Statistics – Number of packets sent	R	Uint64
175F	4	Port 7 Statistics – Number of packets sent	R	Uint64
1763	4	Port 8 Statistics – Number of packets sent	R	Uint64
1767	4	Port 9 Statistics – Number of packets sent	R	Uint64
176B	4	Port 10 Statistics – Number of packets sent	R	Uint64
176F	4	Port 11 Statistics – Number of packets sent	R	Uint64
1773	4	Port 12 Statistics – Number of packets sent	R	Uint64
1777	4	Port 13 Statistics – Number of packets sent	R	Uint64
177B	4	Port 14 Statistics – Number of packets sent	R	Uint64
177F	4	Port 15 Statistics – Number of packets sent	R	Uint64

Address	Number of Registers	Description	R/W	Format
1783	4	Port 16 Statistics – Number of packets sent	R	Uint64
1787	4	Port 17 Statistics – Number of packets sent	R	Uint64
178B	4	Port 18 Statistics – Number of packets sent	R	Uint64
178F	4	Port 19 Statistics – Number of packets sent	R	Uint64
1793	4	Port 20 Statistics – Number of packets sent	R	Uint64
1797	4	Port 21 Statistics – Number of packets sent	R	Uint64
179B	4	Port 22 Statistics – Number of packets sent	R	Uint64
179F	4	Port 23 Statistics – Number of packets sent	R	Uint64
17A3	4	Port 24 Statistics – Number of packets sent	R	Uint64
17A7	4	Port 25 Statistics – Number of packets sent	R	Uint64
17AB	4	Port 26 Statistics – Number of packets sent	R	Uint64
17AF	4	Port 27 Statistics – Number of packets sent	R	Uint64
17B3	4	Port 1 Statistics – Number of bytes received	R	Uint64

Address	Number of Registers	Description	R/W	Format
17B7	4	Port 2 Statistics – Number of bytes received	R	Uint64
17BB	4	Port 3 Statistics – Number of bytes received	R	Uint64
17BF	4	Port 4 Statistics – Number of bytes received	R	Uint64
17C3	4	Port 5 Statistics – Number of bytes received	R	Uint64
17C7	4	Port 6 Statistics – Number of bytes received	R	Uint64
17CB	4	Port 7 Statistics – Number of bytes received	R	Uint64
17CF	4	Port 8 Statistics – Number of bytes received	R	Uint64
17D3	4	Port 9 Statistics – Number of bytes received	R	Uint64
17D7	4	Port 10 Statistics – Number of bytes received	R	Uint64
17DB	4	Port 11 Statistics – Number of bytes received	R	Uint64
17DF	4	Port 12 Statistics – Number of bytes received	R	Uint64
17E3	4	Port 13 Statistics – Number of bytes received	R	Uint64
17E7	4	Port 14 Statistics – Number of bytes received	R	Uint64

Address	Number of Registers	Description	R/W	Format
17EB	4	Port 15 Statistics – Number of bytes received	R	Uint64
17EF	4	Port 16 Statistics – Number of bytes received	R	Uint64
17F3	4	Port 17 Statistics – Number of bytes received	R	Uint64
17F7	4	Port 18 Statistics – Number of bytes received	R	Uint64
17FB	4	Port 19 Statistics – Number of bytes received	R	Uint64
17FF	4	Port 20 Statistics – Number of bytes received	R	Uint64
1803	4	Port 21 Statistics – Number of bytes received	R	Uint64
1807	4	Port 22 Statistics – Number of bytes received	R	Uint64
180B	4	Port 23 Statistics – Number of bytes received	R	Uint64
180F	4	Port 24 Statistics – Number of bytes received	R	Uint64
1813	4	Port 25 Statistics – Number of bytes received	R	Uint64
1817	4	Port 26 Statistics – Number of bytes received	R	Uint64
181B	4	Port 27 Statistics – Number of bytes received	R	Uint64

Address	Number of Registers	Description	R/W	Format
181F	4	Port 1 Statistics – Number of bytes sent	R	Uint64
1823	4	Port 2 Statistics – Number of bytes sent	R	Uint64
1827	4	Port 3 Statistics – Number of bytes sent	R	Uint64
182B	4	Port 4 Statistics – Number of bytes sent	R	Uint64
182F	4	Port 5 Statistics – Number of bytes sent	R	Uint64
1833	4	Port 6 Statistics – Number of bytes sent	R	Uint64
1837	4	Port 7 Statistics – Number of bytes sent	R	Uint64
183B	4	Port 8 Statistics – Number of bytes sent	R	Uint64
183F	4	Port 9 Statistics – Number of bytes sent	R	Uint64
1843	4	Port 10 Statistics – Number of bytes sent	R	Uint64
1847	4	Port 11 Statistics – Number of bytes sent	R	Uint64
184B	4	Port 12 Statistics – Number of bytes sent	R	Uint64
184F	4	Port 13 Statistics – Number of bytes sent	R	Uint64

Address	Number of Registers	Description	R/W	Format
1853	4	Port 14 Statistics – Number of bytes sent	R	Uint64
1857	4	Port 15 Statistics – Number of bytes sent	R	Uint64
185B	4	Port 16 Statistics – Number of bytes sent	R	Uint64
185F	4	Port 17 Statistics – Number of bytes sent	R	Uint64
1863	4	Port 18 Statistics – Number of bytes sent	R	Uint64
1867	4	Port 19 Statistics – Number of bytes sent	R	Uint64
186B	4	Port 20 Statistics – Number of bytes sent	R	Uint64
186F	4	Port 21 Statistics – Number of bytes sent	R	Uint64
1873	4	Port 22 Statistics – Number of bytes sent	R	Uint64
1877	4	Port 23 Statistics – Number of bytes sent	R	Uint64
187B	4	Port 24 Statistics – Number of bytes sent	R	Uint64
187F	4	Port 25 Statistics – Number of bytes sent	R	Uint64
1883	4	Port 26 Statistics – Number of bytes sent	R	Uint64

Address	Number of Registers	Description	R/W	Format
1887	4	Port 27 Statistics – Number of bytes sent	R	Uint64

Interpreting the Port State Value

This section provides information for determining the port state.

In the following table, the upper byte repressents the interface state, and the lower byte represents the line protocol state.

Address	Description	Value
0x1700 to 0x171B	Port state information	Upper byte: • 0x0: Interface is down • 0x1: Interface is going down • 0x2: Interface is in the initializing state • 0x3: Interface is coming up • 0x4: Interface is up and running • 0x5: Interface is reset by the user • 0x6: Interface is shut down by the user • 0x7: Interface is being deleted
		The lower byte: • 0x0: Line protocol state is down • 0x1: Line protocol state is up

Configure MODBUS

The MODBUS TCP server listens for MODBUS client requests on TCP port 502 by default. Port 502 is enabled when MODBUS server is started unless you configure a different port for MODBUS communications. The MODBUS server is disabled by default.

Note MODBUS is supported only on standalone .

To configure MODBUS:

Before you begin

If a firewall or other security services are enabled, the switch TCP port might be blocked, and the switch and the client cannot communicate. If a firewall and other security services are disabled, a denial-of-service attack might occur on the switch. To add security when using MODBUS TCP, configure an ACL to permit traffic from specific clients or configure QoS to rate-limit traffic.

Procedure

Step 1 Enter global configuration mode:

configure terminal

Step 2 Enable MODBUS TCP on the switch:

scada modbus tcp server

To disable MODBUS on the switch and return to the default settings, enter the **no scada modbus tcp server** global configuration command.

The system displays a message to warn you that starting the MODBUS TCP server is a security risk:

WARNING: Starting Modbus TCP server is a security risk. Please understand the security issues involved before proceeding further. Do you still want to start the server? [yes/no]:

Step 3 Enter **yes** to confirm that you understand the security issues and to proceed with starting the server.

Step 4 (Optional) Set the TCP port to which clients send messages:

scada modbus tcp server port *tcp-port-number*

The range for *tcp-port-number* is 1 to 65535. The default is 502.

Step 5 (Optional) Set the number of simultaneous connection requests sent to the switch:

scada modbus tcp server connection *connection-requests*

The range for *connection-requests* is 1 to 5. The default is 1.

Step 6 Return to privileged EXEC mode:

end

Example

```
Switch# configure terminal
Switch(config)# scada modbus tcp server
WARNING: Starting Modbus TCP server is a security risk. Please understand the security
issues involved
before proceeding further. Do you still want to start the server? [yes/no]: y
Switch(config)# end
```

Displaying MODBUS Commands

Use the commands listed below to display information for MODBUS TCP.

Command	Purpose
show scada modbus tcp server	Displays the server information and statistics
show scada modbus tcp server connections	Shows information and statistics for each client connection
clear scada modbus tcp server statistics	Clears all the statistics for the Modbus server, including statistics for each client connection

The following is an example of the **show scada modbus tcp server** command and its output:

```
Switch# show scada modbus tcp server
Summary: enabled, running, process id 142
Conn Stats: listening on port 801, 4 max simultaneous connections
    0 current client connections
    0 total accepted connections, 0 accept connection errors
    0 closed connections, 0 close connection errors
Send Stats: 0 tcp msgs sent, 0 tcp bytes sent, 0 tcp errors
    0 responses sent, 0 exceptions sent, 0 send errors
Recv Stats: 0 tcp msgs received, 0 tcp bytes received, 0 tcp errors
    0 requests received, 0 receive errors
```

PART X

Ring Feature Protocol

CHAPTER 172

Parallel Redundancy Protocol

Information About PRP

From Release 17.17.1, the Cisco IE3505 Rugged Series Switches and IE3505 Heavy-Duty Series Switches supports Parallel Redundancy Protocol (PRP).

Parallel Redundancy Protocol overview

PRP is defined in the International Standard IEC 62439-3. PRP is designed to provide hitless redundancy (zero recovery time after failures) in Ethernet networks.

To recover from network failures, redundancy can be provided by network elements connected in mesh or ring topologies using protocols like RSTP, REP, or MRP, where a network failure causes some reconfiguration in the network to allow traffic to flow again (typically by opening a blocked port). These schemes for redundancy can take between a few milliseconds to a few seconds for the network to recover and traffic to flow again.

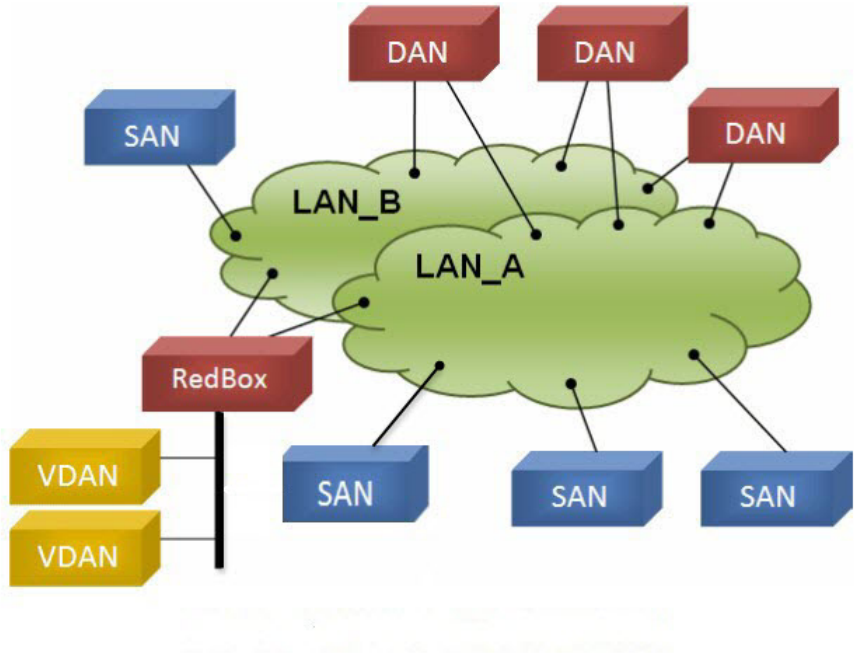
PRP uses a different scheme, where the end nodes implement redundancy (instead of network elements) by connecting two network interfaces to two independent, disjointed, parallel networks (LAN-A and LAN-B). Each of these Dually Attached Nodes (DANs) then have redundant paths to all other DANs in the network.

The DAN sends two packets simultaneously through its two network interfaces to the destination node. A redundancy control trailer (RCT), which includes a sequence number, is added to each frame to help the destination node distinguish between duplicate packets. When the destination DAN receives the first packet successfully, it removes the RCT and consumes the packet. If the second packet arrives successfully, it is discarded. If a failure occurs in one of the paths, traffic continues to flow over the other path uninterrupted, and zero recovery time is required.

Non-redundant endpoints in the network that attach only to either LAN-A or LAN-B are known as Singly Attached Nodes (SANs).

A Redundancy Box (RedBox) is used when an end node that does not have two network ports and does not implement PRP needs to implement redundancy. Such an end node can connect to a RedBox, which provides connectivity to the two different networks on behalf of the device. Because a node behind a RedBox appears for other nodes like a DAN, it is called a Virtual DAN (VDAN). The RedBox itself is a DAN and acts as a proxy on behalf of its VDANs.

Figure 164: PRP Redundant Network



To manage redundancy and check the presence of other DANs, a DAN periodically sends Supervision frames and can evaluate the Supervision frames sent by other DANs.

Switches that Support PRP

Table 184: The following Advanced base modules SKUs (PIDs) supports PRP.

Switch	PID
Cisco IE3505 Rugged Series Switch	IE-3505-8P3S
	IE-3505-8T3S

Switch	PID
Cisco IE3505 Heavy-Duty Series Switch	IE-3505H-16T

Table 185: The following Expansion modules PIDs include PRP support when installed on one of the above advanced base modules

Switch	Expansion modules PID
Cisco IE3505 Rugged Series Switch	IEM-3500-14T2S
	IEM-3500-6T2S
	IEM-3500-16P
	IEM-3500-16T
	IEM-3500-8P
	IEM-3500-8S
	IEM-3500-8T

Support for PRP is available on Network Essentials and Network Advantage licenses.

Supported PRP Features

IE-3505-8P3S, IE-3505-8T3S, and IE-3505H-16T switches support the following PRP features.

PRP supports maximum of two channel instance as shown in the following table:

Table 186: PRP Support for Cisco IE3505 Series Switch and Expansion Models

Switch	FPGA Profile	Number of PRP Instance
Cisco IE3505 Rugged Series without expansion module	Default	1
	Redundancy	2
Cisco IE3505 Rugged Series with expansion module	Default	1
	Redundancy	2
IE-3505H-16T	Default	1
	Redundancy	2

Role of the Switch

The switches implement RedBox functionality using Gigabit Ethernet port connections to each of the two LANs.

PRP Channels

PRP channel or channel group is a logical interface that aggregates two Gigabit Ethernet interfaces (access, trunk, or routed) into a single link. In the channel group, the lower numbered Gigabit Ethernet member port is the primary port and connects to LAN-A. The higher numbered port is the secondary port and connects to LAN-B.

The PRP channel remains up as long as at least one of these member ports remains up and sends traffic. When both member ports are down, the channel is down. The maximum number of supported PRP channel groups per switch is 2, depending on the configured FPGA profile. For information about FPGA profile, see Configure FPGA Profile, on page 2239.

The interfaces that you can use for each group on each switch series are fixed, as shown in the following table.

Table 187: The supported PRP channel interfaces for IE3505 Rugged Series Switch

PRP Channel Number	Module	Ports Interface
PRP Channel 1	Base module with SFP ports	GigabitEthernet1/1 and 1/2
PRP Channel 1	Base module with Copper (CU) ports	GigabitEthernet1/4 and 1/5
PRP Channel 2	Base module with CU ports	GigabitEthernet1/6 and 1/7
PRP Channel 2	8 port CU expansion module	GigabitEthernet2/1 and 2/2
PRP Channel 2	8 port SFP expansion module	GigabitEthernet2/1 and 2/2
PRP Channel 2	16 port CU expansion module	GigabitEthernet2/1 and 2/2
PRP Channel 2	8 port Mix expansion module	GigabitEthernet2/7 and 2/8
PRP Channel 2	16 port Mix expansion module	GigabitEthernet2/15 and 2/16

Table 188: The supported PRP channel interfaces for advanced IE3505 Heavy Duty Series Switch

PRP Channel Number	Module	
PRP Channel 1	Base module with CU ports	GigabitEthernet1/1 and 1/2
PRP Channel 1	Base module with CU ports	GigabitEthernet1/4 and 1/5
PRP Channel 2	Base module with CU ports	GigabitEthernet1/6 and 1/7

Mixed Traffic and Supervision Frames

Traffic egressing the RedBox PRP channel group can be mixed, that is, destined to either SANs (connected only on either LAN-A or LAN-B) or DANs. To avoid duplication of packets for SANs, the switch learns source MAC addresses from received supervision frames for DAN entries and source MAC addresses from non-PRP (regular traffic) frames for SAN entries and maintains these addresses in the node table. When forwarding packets out the PRP channel to SAN MAC addresses, the switch looks up the entry and determines which LAN to send to rather than duplicating the packet.

A RedBox with VDANs needs to send supervision frames on behalf of those VDANs. For traffic coming in on all other ports and going out PRP channel ports, the switch learns source MAC addresses, adds them to the VDAN table, and starts sending supervision frames for these addresses. Learned VDAN entries are subject to aging.

You can add static entries to the node and VDAN tables as described in x. You can also display the node and VDAN tables and clear entries. See y and z.

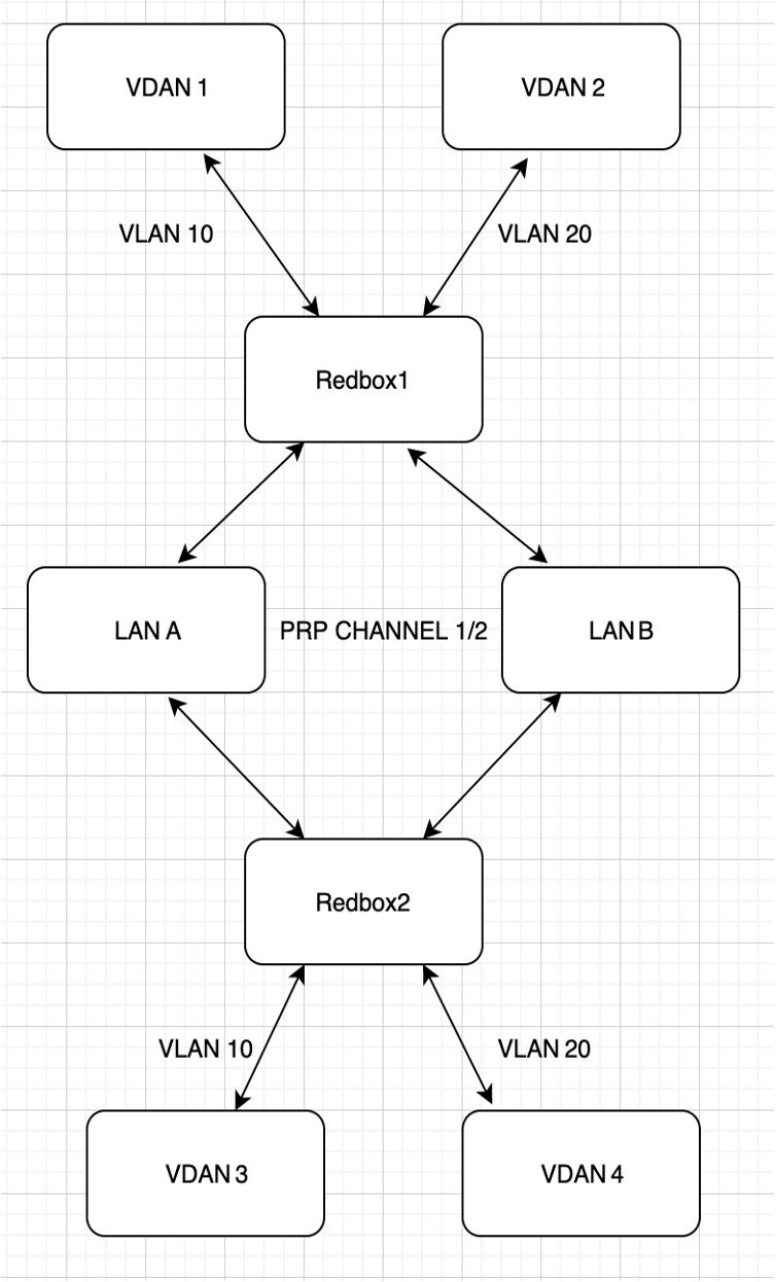
VLAN Tag in Supervision Frame

Switches support VLAN tagging for supervision frames. PRP VLAN tagging requires that PRP interfaces be configured in trunk mode. This feature allows you to specify a VLAN ID in the supervision frames for a PRP channel.

Note The IE3505 Series Switches support 512 VDAN/Node MAC addresses.

The PRP Supervision VLAN-aware feature is not supported in the IE3505 Series Switches.

For information on configuring the VLANs, see Configuring PRP Channel with Supervision Frame VLAN Tagging, on page 2548.



PTP over PRP

PRP provides high availability through redundancy for PTP. For a description of PTP, see *Precision Time Protocol*.

The PRP method of achieving redundancy by parallel transmission over two independent paths does not work for PTP as it does for other traffic. The delay that a frame experiences is not the same in the two LANs, and some frames are modified in the transparent clocks (TCs) while transiting through the LAN. A Dually Attached

Node (DAN) does not receive the same PTP message from both ports even when the source is the same. Specifically:

- Sync/Follow_Up messages are modified by TCs to adjust the correction field.
- Boundary Clocks (BCs) present in the LAN are not PRP-aware and generate their own Announce and Sync frames with no Redundancy Control Trailer (RCT) appended.
- Follow_Up frames are generated by every 2-step clock and carry no RCT.
- TCs are not PRP-aware and not obliged to forward the RCT, which is a message part that comes after the payload.

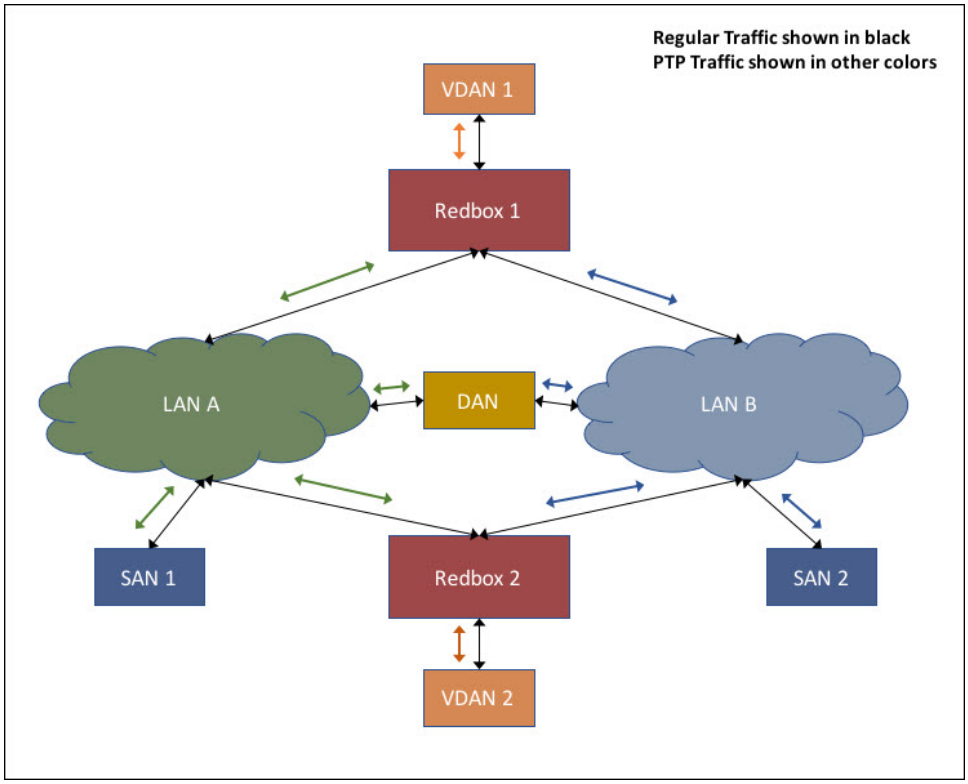
Before support of PTP over LAN-A and LAN-B, PTP traffic was allowed only on LAN-A to avoid the issues with PTP and parallel transmission described earlier. However, if LAN-A went down, PTP synchronization was lost. To enable PTP to leverage the benefit of redundancy offered by the underlying PRP infrastructure, PTP packets over PRP networks are handled differently than other types of traffic.

The implementation of the PTP over PRP feature is based on the PTP over PRP operation that is detailed in IEC 62439-3:2016, *Industrial communication networks - High availability automation networks - Part 3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR)*. This approach overcomes the problems mentioned earlier by not appending an RCT to PTP packets and bypassing the PRP duplicate/discard logic for PTP packets.

PTP over PRP Packet Flow

The following figure illustrates the operation of PTP over PRP.

Figure 165: PTP over PRP Packet Flow



In the figure, VDAN 1 is the grandmaster clock (GMC). Dually attached devices receive PTP synchronization information over both their PRP ports. The LAN-A port and LAN-B port use a different virtual clock that is synchronized to the GMC. However, only one of the ports (referred to as time recipient) is used to synchronize the local clock (VDAN 2 in the figure). While the LAN-A port is the time recipient, the LAN-A port's virtual clock is used to synchronize VDAN-2. The other PRP port, LAN-B, is referred to as PASSIVE. The LAN-B port's virtual clock is still synchronized to the same GMC, but is not used to synchronize VDAN 2.

If LAN-A goes down, the LAN-B port takes over as the time recipient and is used to continue synchronizing the local clock on RedBox 2. VDAN 2 attached to RedBox 2 continues to receive PTP synchronization from RedBox 2 as before. Similarly, all DANs, VDANs, and RedBoxes shown in the figure continue to remain synchronized. For SANs, redundancy is not available, and in this example, SAN 1 loses synchronization if LAN-A goes down.

Due to the change, VDAN 2 may experience an instantaneous shift in its clock due to the offset between the LAN-A port's virtual clock and the LAN-B port's virtual clock. The magnitude of the shift should only be a few microseconds at the most, because both clocks are synchronized to the same GMC. The shift also occurs when the LAN-A port comes back as time recipient and the LAN-B port becomes PASSIVE.

Note Cisco is moving from the traditional Master/Slave nomenclature. In this document, the terms Grandmaster clock (GMC) or time source and time recipient are used instead, where possible. Exceptions may be present due to language that is hard-coded in the user interfaces of the product software, language used based on RFP documentation, or language that is used by a referenced third-party product.

Supported Location of GMC

The GMC can be located in a PTP over PRP topology as one of the following:

- A RedBox that is connected to both LAN A and LAN B (for example, RedBox 1 in the preceding diagram).
- A VDAN (for example, VDAN 1 in the preceding diagram).
- A DAN (for example, the DAN in the preceding diagram).

The GMC cannot be a SAN attached to LAN-A or LAN-B, because only the devices in LAN-A or LAN-B will be synchronized to the GMC.

Configuration

PTP over PRP does not require configuration beyond how you would normally configure PTP and PRP separately, and there is no user interface added for this feature. The difference is that before the PTP over PRP feature, PTP worked over LAN-A only; now it works over both LANs. Before implementing PTP over PRP, refer to Guidelines and Limitations.

The high-level workflow to implement PTP over PRP in your network is as follows:

1. Refer to the section RedBox Types in this guide to determine the location of the PRP RedBox. Refer to *Precision Time Protocol* to determine the PTP mode and profile.
2. Configure PTP as described in *Precision Time Protocol* and follow the procedure for the PTP profile determined in step 1.
3. Configure PRP as described in Create a PRP Channel and Group.

Supported PTP Profiles and Clock Modes

The following table summarizes PTP over PRP support for the various PTP profiles and clock modes. In unsupported PTP profile/clock mode combinations, PTP traffic flows over LAN-A only. LAN-A is the lower numbered interface. See PRP Channels for PRP interface numbers.

PTP Profile	Clock Mode	Supported?	PRP RedBox type as per IEC 62439-3
End-to-End Delay Request-Response default profile	BC	Yes	PRP RedBox as doubly attached BC (DABC) with E2E
	E2E TC	No	PRP RedBox as doubly attached TC (DATC) with E2E
Power Profile	BC	Yes	PRP RedBox as doubly attached BC (DABC) with P2P
	P2P TC	Yes	PRP RedBox as doubly attached TC (DATC) with P2P

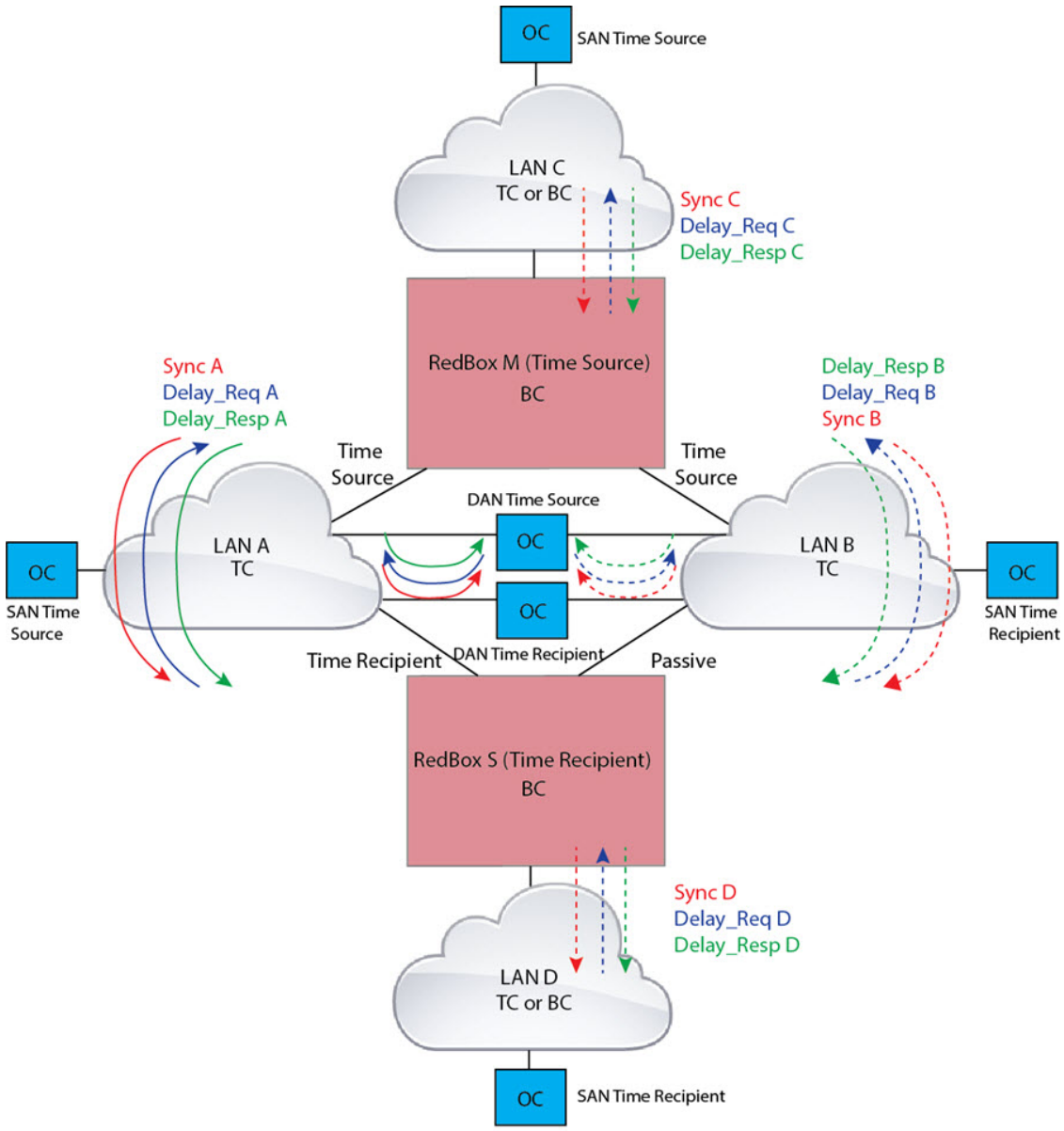
PRP RedBox Types

The switch plays the role of a RedBox in PRP networks. This section describes the types of PRP RedBoxes supported for PTP over PRP as defined in IEC 62439-3.

PRP RedBox as a Doubly Attached BC (DABC) with E2E

In the configuration shown below, two RedBoxes (for example, M and S) are configured as Boundary Clocks (BCs) that use the End-to-End delay measurement mechanism and IEEE1588v2 Default Profile. The Best Master Clock Algorithm (BMCA) on RedBox M determines port A and port B to be connected to the time source. The PTP protocol running on Redbox M treats both ports A and B individually as time source ports and sends out Sync and Follow_Up messages individually on both the ports.

Figure 166: PRP Redbox as DABC with E2E



On Redbox S, the regular BMCA operation determines port A to be a time recipient and port B to be PASSIVE. However, with the knowledge that ports A and B are part of the same PRP channel, port B is forced into PASSIVE_SLAVE state. Port A and Port B on Redbox S operate as follows:

- Port A works as a regular time recipient port. It uses the end-to-end delay measurement mechanism to calculate delay and offset from the time source. Using the calculated delay and offset, it synchronizes the local clock.
- Port B is in PASSIVE_SLAVE state. It uses the end-to-end delay measurement mechanism to calculate delay and offset from the time source.

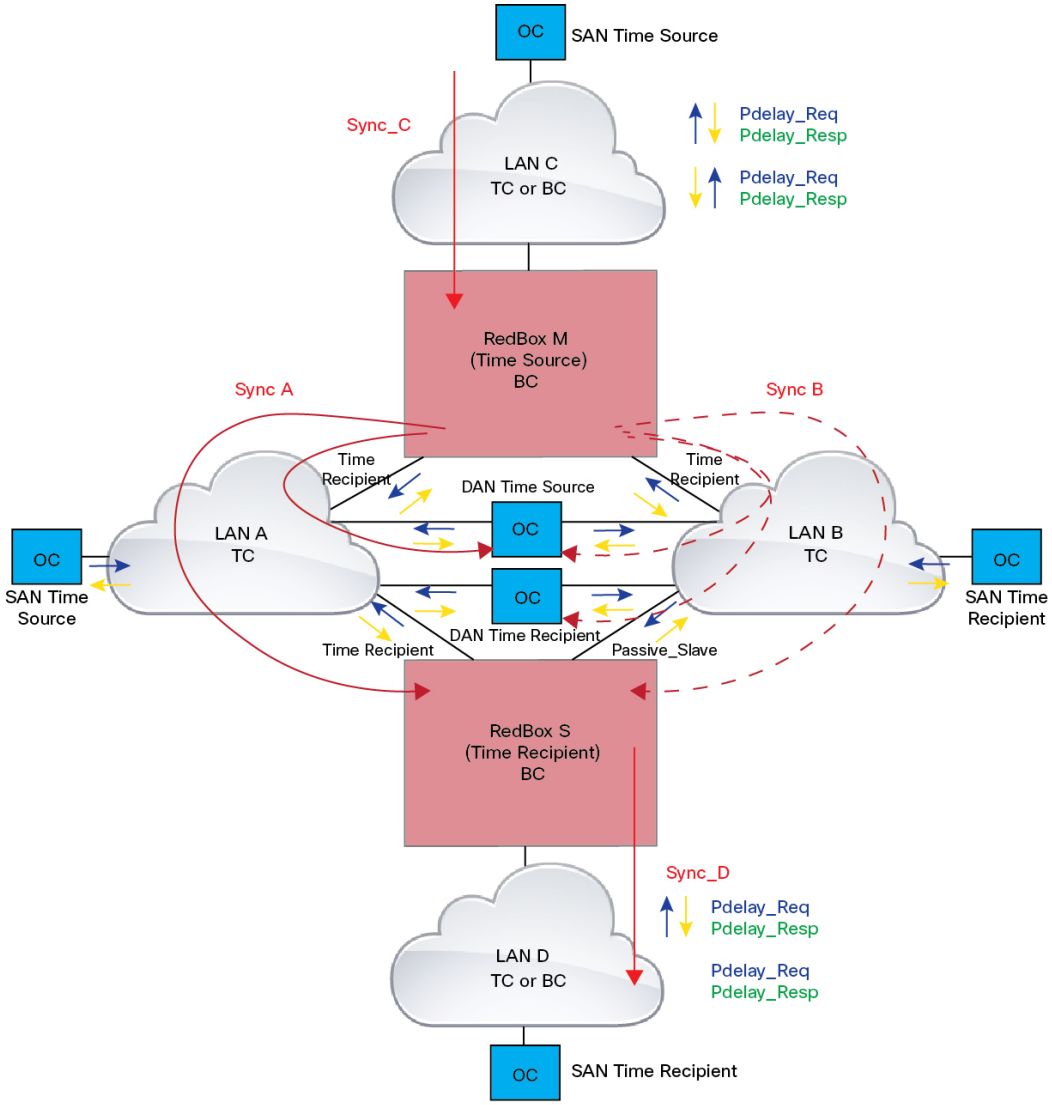
It is passive in the sense that it maintains the calculated delay and offset, but does not perform any operation on the local clock. Having the delay and offset information readily available equips it to seamlessly change its role to time recipient if there is loss of connectivity to the time source on port A.

PRP RedBox as Doubly Attached BC (DABC) with P2P

The following figure shows an example where Redbox M and Redbox S are configured to run in Power Profile as Boundary Clocks that use Peer-to-Peer (P2P) delay measurement mechanism. In this example, the GMC is the ordinary clock attached through LAN C. All the clocks are configured to run Peer-to-Peer Delay measurement and the peer delay is regularly calculated and maintained on every link shown in the figure.

The BMCA on Redbox M determines ports A and B to be connected to the time source. The PTP protocol running on Redbox M treats both ports A and B individually as time source ports and sends out Sync and Follow_Up messages individually on both the ports.

Figure 167: PRP Redbox as DABC with P2P



On Redbox S, the regular BMCA operation determines port A to be time recipient and port B to be PASSIVE. However, with the knowledge that ports A and B are part of the same PRP channel, port B is forced into PASSIVE_SLAVE state. Port A and Port B on Redbox S operate as follows:

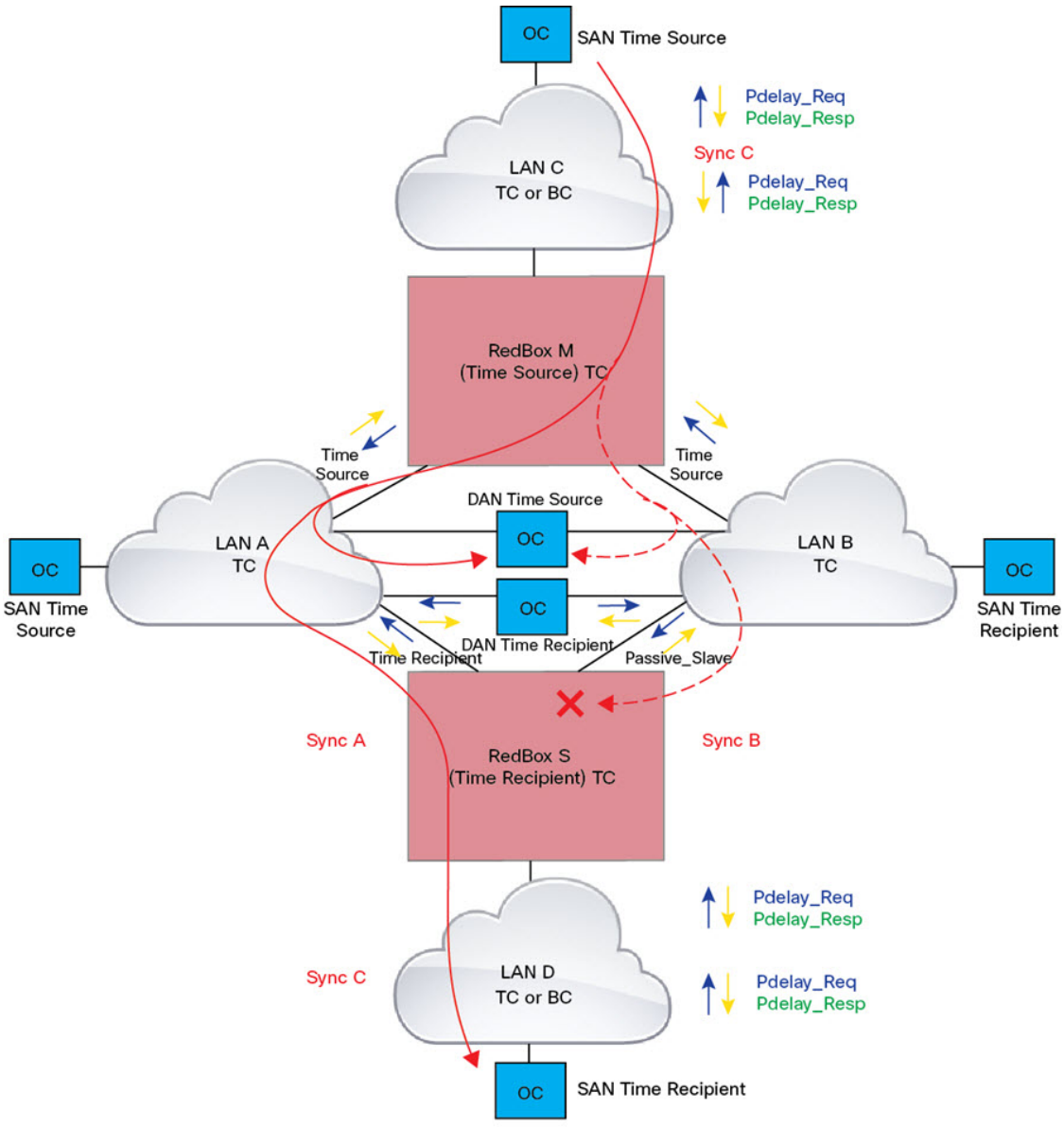
- Port A works as a regular time recipient port. It uses the Sync and Follow_Up messages along with their correction field to calculate the delay and offset from time source and synchronize the local clock. (Unlike an E2E BC, it does not need to generate Delay_Req messages because all the link delays and residence times along the PTP path are accumulated in the correction field of the Follow_Up messages).
- Port B is in PASSIVE_SLAVE state. Like port A, it maintains the delay and offset from time source, but does not perform any operation on the local clock. Having all the synchronization information available enables it to seamlessly take over as the new time recipient in case port A loses communication with the GM.

PRP RedBox as Doubly Attached TC (DATC) with P2P

The following figure shows an example where Redbox M and Redbox S are configured to run in Power Profile mode as Transparent Clocks. In this example, the GMC is the ordinary clock attached through LAN C. All the clocks are configured to run Peer-to-Peer Delay measurement and the peer delay is regularly calculated and maintained on every link shown in the figure.

Redbox M and Redbox S run BMCA even though it is not mandatory for a P2P TC to run BMCA. On Redbox M, the BMCA determines ports A and B to be connected to the time source. Redbox M forwards all Sync and Follow_Up messages received on port C out of ports A and B.

Figure 168: PRP Redbox as DATC with P2P



On Redbox S, port A is determined to be time recipient and port B to be PASSIVE_SLAVE as described earlier. Port A and Port B on Redbox S operate as follows:

- Port A works as a regular time recipient port. It uses the Sync and Follow_Up messages along with their correction field to calculate the delay and offset from time source and synchronize the local clock. (Unlike an E2E BC, it does not need to generate Delay_Req messages since all the link delays and residence times along the PTP path are accumulated in the correction field of the Follow_Up messages).
- Like port A, port B maintains the delay and offset from time source, but does not perform any operation on the local clock. Having all the synchronization information available enables it to seamlessly take over as the new time recipient in case port A loses communication with the GMC.

LAN-A and LAN-B Failure Detection and Handling

Failures in LAN-A and LAN-B are detected and handled in the same way for all RedBox types that are described in PRP RedBox Types.

Using the example that is shown in PRP RedBox as DATC with P2P with the GMC as a SAN in LAN C, a failure in LAN-A or LAN-B pertaining to PTP can occur due to the following reasons:

- A device within the LAN goes down.
- A link within the LAN goes down resulting in loss of connectivity.
- PTP messages are dropped within the LAN.

These events result in PTP Announce Receipt Timeout on RedBox S, which triggers the BMCA calculation. Refer to section 7.7.3.1 of the IEEE 1588v2 standard for details on Announce Receipt Timeout.

The BMCA, once invoked, changes the state of the PASSIVE_SLAVE port to time recipient and time recipient to PASSIVE_SLAVE or PASSIVE or FAULTY. The state changes are done atomically to avoid transient cases where there are two time recipient ports or two PASSIVE_SLAVE ports.

RedBox S now synchronizes to the GMC over the new time recipient port. The change to synchronization should be quick and seamless, unless the delays experienced by PTP packets on the two LANs are very different or if there are some non-PTP devices in the LANs.

The SAN time recipient in LAN D also sees this shift in the timing from RedBox S and must converge to the new clock. This is similar to a GMC change event for this clock, but as mentioned earlier, the change is usually seamless.

TrustSec on a PRP Interface

You can configure Cisco TrustSec (CTS) on member interfaces of a PRP channel.

Because TrustSec is supported only on physical interfaces, you cannot configure TrustSec on the logical PRP channel interface. A PRP channel includes two interfaces, for example, Gi1/1 and Gi1/2. To configure TrustSec on interfaces that are members of a PRP channel, ensure that the following conditions are met:

- The Network Advantage license is required to use TrustSec.
- Configure TrustSec on each interface first, before it is part of the PRP channel.
- The TrustSec configuration on both PRP channel interfaces must be the same to allow inline tagging and propagation with LAN-A and LAN-B as expected.

Note CTS + Security Association Protocol (SAP) and CTS + MACsec Key Agreement (MKA) methods are not supported over PRP interface.

Configuring TrustSec on a PRP Interface

This section provides examples for configuring TrustSec on a PRP interface. You can configure the PRP channel interfaces by configuring each individual interface or by using the **interface range** <>.

Valid Configuration

The following example shows configuring TrustSec on each interface, one at a time, and then making that individual interface part of a PRP channel.

```
switch#configure terminal
switch(config)#int gi1/1
switch(config-if)#switchport mode access
switch(config-if)#switchport access vlan 30
switch(config-if)#cts manual
switch(config-if-cts-manual)#policy static sgt 1000 trusted
switch(config-if-cts-manual)#exit
switch(config-if)#prp-channel-group 1
Creating a PRP-channel interface PRP-channel 1

switch(config-if)#
switch(config-if)#int gi1/2
switch(config-if)#switchport mode access
switch(config-if)#switchport access vlan 30
switch(config-if)#cts manual
switch(config-if-cts-manual)#policy static sgt 1000 trusted
switch(config-if-cts-manual)#exit
switch(config-if)#prp-channel-group 1
switch(config-if)#end
```

The following example shows configuring TrustSec on a range of interfaces and then making the interfaces part of a PRP channel.

```
switch#configure terminal
switch(config)#int range gi1/1-2
switch(config-if-range)#switchport mode access switch
switch(config-if-range)#switchport access vlan 30
switch(config-if-range)#cts manual
switch(config-if-cts-manual)#policy static sgt 1000 trusted
switch(config-if-cts-manual)#exit
switch(config-if-range)#prp-channel-group 1
Creating a PRP-channel interface PRP-channel 1
```

CTS and PRP Show Commands

This section lists **show** commands that you can use when configuring TrustSec on PRP member interfaces and examples of some command outputs:

- **show cts interface summary**
- **show cts pacs**

- **show cts interface <>**
- **show cts role-based counters**
- **show prp channel detail**
- **show prp statistics ingressPacketStatistics**
- **show prp statistics egressPacketStatistics**

The following example show the output of the **show cts interface summary** command:

```
switch#show cts interface summary
CTS Interfaces
----------------------
Interface                       Mode    IFC-state dot1x-role peer-id    IFC-cache
Critical-Authentication
------------------------------------------------------------------------------
Gi1/1                           MANUAL  OPEN      unknown    unknown    invalid  Invalid
Gi1/2                           MANUAL  OPEN      unknown    unknown    invalid  Invalid

R1#show cts pacs
AID: 51F577DCE176855650F2F5609418AC6
PAC-Info:
  PAC-type = Cisco Trustsec
  AID: 51F577DC7E176855650F2F5609418AC6
  I-ID: petra3400ipv4
  A-ID-Info: Identity Services Engine
  Credential Lifetime: 09:06:08 UTC Wed Nov 01 2023
PAC-Opaque:
000200B80003000100040001051F577DC7E176855650F2F5609418AC60006009C000301002BBB79441FEE97B0E0B339B9036F9C7100000001364C8D
1A000093A8054BC5FA1780A24E23B60A4BFF46AF47A317EE20391BFCA6F0CAABA7F66393F05799A3B0EAB602B54749DCF7225A45FDDB1349A81977D857B9C3
1959A2B54CFC4505CD903D84394E69E5795D31543BB575FB8D51A6FA021FB5E6A0C296F8CA21318377688073516714125D38973D9EF2A66792E3AD1C0A05C3
E739CA1
Refresh timer is set for 12w4d
R1#show cts interface GigabitEthernet1/1
Global Dot1x feature is Disabled
Interface GigabitEthernet1/1:
    CTS is enabled, mode:    MANUAL
    IFC state:               OPEN
    Interface Active for 00:03:25.772
    Authentication Status:   NOT APPLICABLE
        Peer identity:       "unknown"
        Peer's advertised capabilities: ""
    Authorization Status:    SUCCEEDED
        Peer SGT:            30
        Peer SGT assignment: Trusted
    SAP Status:              NOT APPLICABLE
    Propagate SGT:           Enabled
    Cache Info:
        Expiration            : N/A
        Cache applied to link : NONE

    Statistics:
        authc success:              0
        authc reject:               0
        authc failure:              0
        authc no response:          0
        authc logoff:               0
        sap success:                0
        sap fail:                   0
        authz success:              0
        authz fail:                 0
        port auth fail:             0
```

```
    L3 IPM:   disabled.
```

The following example shows the output of the **show cts role-based counters** command:

```
switch# show cts role-based counters
Role-based IPv4 counters
From    To      SW-Denied  HW-Denied  SW-Permitt HW-Permitt SW-Monitor
HW-Monitor
*       *       0          0          0          0          0          0
122     0       0          0          0          0          0          0
200     0       0          0          0          2845       0          0
201     130     0          0          0          0          0          0
130     200     0          0          0          2845       0          0
```

The following example shows the output of the **show prp channel detail** command:

```
switch#show prp channel 1 summary
Flags:  D - down        P - bundled in prp-channel
        R - Layer3      S - Layer2
        U - in use

Number of channel-groups in use: 1
Group  PRP-channel   Ports
------+-------------+---------------------------------------
1      PR1(SU)       Gi1/1(P), Gi1/2(P)

R1#show prp channel 1 detail
PRP-channel: PR1
------------
 Layer type = L2
 Ports: 2 Maxports = 2
 Port state = prp-channel is Inuse
 Protocol = Enabled
Ports in the group:
  1) Port: Gi1/1
   Logical slot/port = 1/1 Port state = Inuse
 Protocol = Enabled
  2) Port: Gi1/2
   Logical slot/port = 1/2 Port state = Inuse
 Protocol = Enabled
```

The following example shows the output of the **show prp statistics ingressPacketStatistics** command:

```
switch#sh prp statistics ingressPacketStatistics
 PRP prp_maxchannel 2 INGRESS STATS:
 PRP channel-group 1 INGRESS STATS:
   ingress pkt lan a: 1010
   ingress pkt lan b: 1038
   ingress crc lan a: 0
   ingress crc lan b: 0
   ingress danp pkt acpt: 20
   ingress danp pkt dscrd: 20
   ingress supfrm rcv a: 382
   ingress supfrm rcv b: 390
   ingress over pkt a: 0
   ingress over pkt b: 0
   ingress pri over pkt a: 0
   ingress pri over pkt b: 0
   ingress oversize pkt a: 0
   ingress oversize pkt b: 0
```

```
   ingress byte lan a: 85127
   ingress byte lan b: 85289
   ingress wrong lan id a: 402
   ingress wrong lan id b: 402
   ingress warning lan a: 1
   ingress warning lan b: 1
   ingress warning count lan a: 137
   ingress warning count lan b: 137
   ingress unique count a: 0
   ingress unique count b: 0
   ingress duplicate count a: 20
   ingress duplicate count b: 20
   ingress multiple count a: 0
   ingress multiple count b: 0

PRP channel-group 2 INGRESS STATS:
   ingress pkt lan a: 0
   ingress pkt lan b: 0
   ingress crc lan a: 0
   ingress crc lan b: 0
   ingress danp pkt acpt: 0
   ingress danp pkt dscrd: 0
   ingress supfrm rcv a: 0
   ingress supfrm rcv b: 0
   ingress over pkt a: 0
   ingress over pkt b: 0
   ingress pri over pkt a: 0
   ingress pri over pkt b: 0
   ingress oversize pkt a: 0
   ingress oversize pkt b: 0
   ingress byte lan a: 0
   ingress byte lan b: 0
   ingress wrong lan id a: 0
   ingress wrong lan id b: 0
   ingress warning lan a: 0
   ingress warning lan b: 0
   ingress warning count lan a: 0
   ingress warning count lan b: 0
   ingress unique count a: 0
   ingress unique count b: 0
   ingress duplicate count a: 0
   ingress duplicate count b: 0
   ingress multiple count a: 0
   ingress multiple count b: 0
```

The following example shows the output of the **show prp statistics egressPacketStatistics** command:

```
switch#sh prp statistics egressPacketStatistics
 PRP channel-group 1 EGRESS STATS:
   duplicate packet: 20
   supervision frame sent: 427
   packet sent on lan a: 934
   packet sent on lan b: 955
   byte sent on lan a: 96596
   byte sent on lan b: 96306
   egress packet receive from switch: 517
   overrun pkt: 0
   overrun pkt drop: 0
 PRP channel-group 2 EGRESS STATS:
   duplicate packet: 0
   supervision frame sent: 0
   packet sent on lan a: 0
   packet sent on lan b: 0
```

```
byte sent on lan a: 0
byte sent on lan b: 0
egress packet receive from switch: 0
overrun pkt: 0
overrun pkt drop: 0
```

TrustSec Debugging Commands

This section lists **debug** commands that you can use when troubleshooting TrustSec on PRP member interfaces.

- **debug prp errors**
- **debug prp events**
- **debug prp detail**
- **debug cts error**
- **debug cts aaa**
- **debug cts all**

Prerequisites

Network Essentials or Network Advantage License

Guidelines and Limitations

Guidelines

- Because PRP DANs and RedBoxes add a 6-byte PRP trailer to the packet, PRP packets can be dropped by some switches with a maximum transmission unit (MTU) size of 1500. To ensure that all packets can flow through the PRP network, increase the MTU size for switches within the PRP LAN-A and LAN-B network to 1506 as follows: **system mtu 1506**.
- To configure supervision frame VLAN tagging, you must configure interfaces in trunk mode.

Note You cannot configure access mode on PRP interfaces when supervision frame vlan tag configuration exists. If you attempt to configure access mode on a PRP interface with supervision frame VLAN tagging, the system displays this message:

```
%PRP_MSG-4-PRP_VLANTAG: Warning: Do not configure access
mode for PRP interfaces with tagged supervision frames.
```

- A PRP channel must have two active ports that are configured within a channel to remain active and maintain redundancy.
- Both interfaces within a channel group must have the same configuration.

- For Layer 3, you must configure the IP address on the PRP channel interface.
- LLDP and CDP must be disabled on interfaces where PRP is enabled.
- UDLD must be disabled on interfaces where PRP is enabled, especially if the interfaces have media-type sfp.
- The **spanning-tree bpdufilter enable** command is required on the prp-channel interface. Spanning-tree BPDU filter drops all ingress/egress BPDU traffic. This command is required to create independent spanning-tree domains (zones) in the network.
- The **spanning-tree portfast edge trunk** command is optional on the prp-channel interface but highly recommended. It improves the spanning-tree converge time in PRP LAN-A and LAN-B.
- For PRP statistics, use the **show interface prp-channel** [**1** | **2**] command. Physical interface show commands, such as **show interface gi1/1**, do not provide PRP statistics information.
- For switches, use the **int Gi1/1** or **int Gi1/2**, as shown in the following example:

```
switch(config)#int Gi1/1
switch(config-if)#shut
%Interface GigabitEthernet1/1 is configured in PRP-channel group, shutdown not permitted!
```

- PRP functionality can be managed using the CIP protocol. The following CIP commands for PRP are available on:
  - show cip object prp <*0-2*>
  - show cip object nodetable <*0-2*>

Limitations

- PRP traffic load cannot exceed 90 percent bandwidth of the Gigabit Ethernet interface channels.
- Load-balancing is not supported.
- The Protocol status displays incorrectly for the Layer type = L3 section when you enter the **show prp channel detail** command. Refer to the Ports in the group section of the output for the correct Protocol status.

```
show prp channel detail
PRP-channel listing:
                ---------------------

PRP-channel: PR1
------------
 Layer type = L2
 Ports: 2       Maxports = 2
 Port state = prp-channel is Inuse
 Protocol = Enabled
Ports in the group:
  1) Port: Gi1/1
   Logical slot/port = 1/1      Port state = Inuse
        Protocol = Enabled
  2) Port: Gi1/2
   Logical slot/port = 1/2      Port state = Inuse
        Protocol = Enabled

PRP-channel: PR2
```

```
------------
 Layer type = L2
 Ports: 2       Maxports = 2
 Port state = prp-channel is Inuse
 Protocol = Enabled
Ports in the group:
  1) Port: Gi1/6
   Logical slot/port = 1/6      Port state = Inuse
        Protocol = Enabled
  2) Port: Gi1/7
   Logical slot/port = 1/7      Port state = Inuse
        Protocol = Enabled
```

- When an individual PRP interface goes down, **show interface status** continues to show a status of UP for the link. This is because the port status is controlled by the PRP module. Use the **show prp channel** command to confirm the status of the links, which will indicate if a link is down.

  The following example shows the output for the **show prp channel** command:

```
show prp channel 1 detail

PRP-channel: PR1
------------
 Layer type = L2
 Ports: 2       Maxports = 2
 Port state = prp-channel is Inuse
 Protocol = Enabled
Ports in the group:
  1) Port: Gi1/1
   Logical slot/port = 1/1      Port state = Inuse
        Protocol = Enabled
  2) Port: Gi1/2
   Logical slot/port = 1/2      Port state = Inuse
        Protocol = Enabled
```

Node and VDAN Tables

- The switch supports up to 512 (SAN+DANP) entries in the node table.
- The maximum static Node/VDAN count is 16.
- Hash collisions can limit the number of MAC addresses. If the node table is out of resources for learning a MAC address from a node, the switch will default to treating that node as a DAN.
- After reload (before any MAC address is learned), the switch will temporarily treat the unlearned node as a DAN and duplicate the egress packets until an ingress packet or supervision frame is received from the node to populate an entry into the node table.
- The switch supports up to 512 VDAN entries in the VDAN table. If the VDAN table is full, the switch cannot send supervision frames for new VDANS.

Default Settings

By default, no PRP channel exists on the switch until you create it. Interfaces that can be configured for PRP are fixed, as described in PRP Channels, on page 2528.

Create a PRP Channel and Group

To create and enable a PRP channel and group on the switch, follow these steps:

Before you begin

- Review the specific interfaces supported for each switch type, described in PRP Channels, on page 2528.
- Review the Prerequisites, on page 2543 and Guidelines and Limitations, on page 2543.
- Ensure that the member interfaces of a PRP channel are not participating in any redundancy protocols such as FlexLinks, EtherChannel, or REP, before creating a PRP channel.

Procedure

Step 1 Enter global configuration mode:

configure terminal

Step 2 Assign two Gigabit Ethernet interfaces to the PRP channel group. For channel 1, enter:

interface range gigabitethernet 1/1-2

For channel 2, enter:

interface range gigabitethernet 1/6-7

Use the **no interface prp-channel 1|2** command to disable PRP on the defined interfaces and shut down the interfaces.

Note
You must apply the Gi1/1 interface before the Gi1/2 interface. We recommend using the **interface range** command. Similarly, you must apply the Gi1/6 interface before the Gi1/7 for PRP channel 2.

Step 3 (Optional) For Layer 2 traffic, enter **switchport**. (Default):

switchport

Note
For Layer 3 traffic, enter **no switchport**.

Step 4 (Optional) Set a nontrunking, nontagged single VLAN Layer 2 (access) interface:

switchport mode access

Step 5 (Optional) Create a VLAN for the Gigabit Ethernet interfaces:

switchport access vlan <*value*>

Note
This step is required only for Layer 2 traffic.

Step 6 (Optional) Disable Precision Time Protocol (PTP) on the switch:

no ptp enable

PTP is enabled by default. You can disable it if you do not need to run PTP.

Step 7 Disable loop detection for the redundancy channel:

no keepalive

Step 8 Disable UDLD for the redundancy channel:

udld port disable

Step 9 Enter subinterface mode and create a PRP channel group:

prp-channel-group *prp-channel group*

prp-channel group: Value of 1 or 2

The two interfaces that you assigned in step 2 are assigned to this channel group.

The **no** form of this command is not supported.

Step 10 Bring up the PRP channel:

no shutdown

Step 11 Specify the PRP interface and enter interface mode:

interface prp-channel *prp-channel-number*

prp-channel-number: Value of 1 or 2

Step 12 Configure bpdufilter on the prp-channel interface:

spanning-tree bpdufilter enable

The spanning-tree BPDU filter drops all ingress and egress BPDU traffic. This command is required to create independent spanning-tree domains (zones) in the network.

Step 13 (Optional) Configure LAN-A/B ports to quickly get to FORWARD mode:

spanning-tree portfast edge trunk

This command is optional but highly recommended. It improves the spanning-tree convergence time on PRP RedBoxes and LAN-A and LAN-B switch edge ports. It is also highly recommended to configure this command on the LAN_A/LAN_B ports that are directly connected to a RedBox PRP interface.

Examples

The following example shows how to create a PRP channel, create a PRP channel group, and assign two ports to that group.

```
switch# configure terminal
switch(config)# interface range GigabitEthernet1/1-2
switch(config-if)# no keepalive
switch(config-if)# prp-channel-group 1
switch(config-if)# no shutdown
switch(config-if)# exit
```

```
switch(config)# interface prp-channel 1
switch(config)# spanning-tree bpdufilter enable
```

This example shows how to create a PRP channel with a VLAN ID of 2.

```
switch# configure terminal
switch(config)# interface range GigabitEthernet1/1-2
switch(config-if)# switchport
switch(config-if)# switchport mode access
switch(config-if)# switchport access vlan 2
switch(config-if)# no ptp enable
switch(config-if)# no keepalive
switch(config-if)# udld port disable
switch(config-if)# prp-channel-group 1
switch(config-if)# no shutdown
switch(config-if)# exit
switch(config)# interface prp-channel 1
switch(config)# spanning-tree bpdufilter enable
```

This example shows how to create a PRP channel on a switch configured with Layer 3.

```
switch# configure terminal
switch(config)# interface range GigabitEthernet1/1-2
switch(config-if)# no switchport
switch(config-if)# no ptp enable
switch(config-if)# no keepalive
switch(config-if)# udld port disable
switch(config-if)# prp-channel-group 1
switch(config-if)# no shutdown
switch(config-if)# exit
switch(config)# interface prp-channel 1
switch(config)# spanning-tree bpdufilter enable
switch(config)# ip address 192.0.2.10 255.255.255.0
```

Configuring PRP Channel with Supervision Frame VLAN Tagging

To create and enable a PRP channel and group on the switch with VLAN-tagged supervision frames, follow these steps:

Before you begin

- Review the specific interfaces supported for each switch type, as described in PRP Channels, on page 2528.
- Review the Prerequisites, on page 2543 and Guidelines and Limitations, on page 2543.
- Ensure that the member interfaces of a PRP channel are not participating in any redundancy protocols such as FlexLinks, EtherChannel, REP, and so on before creating a PRP channel.

Procedure

Step 1 Enter global configuration mode:

configure terminal

Step 2 Assign two Gigabit Ethernet interfaces to the PRP channel group. For channel 1, enter:

interface gigabitethernet 1/1-2

For channel 2, enter:

interface gigabitethernet 1/6-7

Use the **no interface prp-channel 1|2** command to disable PRP on the defined interfaces and shut down the interfaces.

Note
You must apply the Gi1/1 interface before the Gi1/2 interface. So, we recommend using the **interface range** command.

Step 3 Configure the PRP interface for trunk administrative mode, to allow the interface to carry traffic for more than one VLAN.

switchport mode trunk

Step 4 Specify the allowed VLANS for the trunk interface:

switchport trunk allowed vlan *value*

value: Allowed VLAN number from 0 to 4095 or list of VLANs separated by commas.

Step 5 (Optional) Disable Precision Time Protocol (PTP) on the switch:

no ptp enable

PTP is enabled by default. You can disable it if you do not need to run PTP.

Step 6 Disable loop detection for the redundancy channel:

no keepalive

Step 7 Disable UDLD for the redundancy channel:

udld port disable

Step 8 Enter sub-interface mode and create a PRP channel group:

prp-channel-group *prp-channel group*

prp-channel group: Value of 1 or 2

The two interfaces that you assigned in step 2 are assigned to this channel group.

The **no** form of this command is not supported.

Step 9 Bring up the PRP channel:

no shutdown

Step 10 Specify the PRP interface and enter interface mode:

interface prp-channel *prp-channel-number*

prp-channel-number: Value of 1 or 2

Step 11 Configure bpdufilter on the prp-channel interface:

spanning-tree bpdufilter enable

Spanning-tree BPDU filter drops all ingress/egress BPDU traffic. This command is required to create independent spanning-tree domains (zones) in the network.

Step 12 Set the VLAN ID to be used in VLAN tags for supervision frames:

prp channel-group *prp-channel-number* **supervisionFrameOption vlan-id** *value*

prp-channel-number: Value of 1 or 2

value: VLAN number from 0 to 4095

Step 13 (Optional) Configure the Class of Service (COS) value to be set in the VLAN tag of the Supervision frame:

prp channel-group *prp-channel-number* **supervisionFrameOption vlan-cos** *value*

value: Range is 1 to 7. The default is 1.

Step 14 Enable VLAN tagging on the interface:

prp channel-group *prp-channel-number* **supervisionFrameOption vlan-tagged** *value*

prp-channel-number: Value of 1 or 2

Step 15 (Optional) Configure LAN-A/B ports to quickly get to FORWARD mode:

spanning-tree portfast edge trunk

This command is optional but highly recommended. It improves the spanning-tree convergence time on PRP RedBoxes and LAN-A and LAN-B switch edge ports. It is also highly recommended to configure this command on the LAN_A/LAN_B ports directly connected to a RedBox PRP interface.

Example

```
REDBOX1# configure terminal
REDBOX1(config)#int range GigabitEthernet1/1-2
REDBOX1(config-if-range)#switchport mode trunk
REDBOX1(config-if-range)#switchport trunk allowed vlan 10,20
REDBOX1(config-if-range)#no ptp enable
REDBOX1(config-if-range)#no keepalive
REDBOX1(config-if-range)#udld port disable
REDBOX1(config-if-range)#no shutdown
REDBOX1(config-if-range)#prp-channel-group 1
REDBOX1(config-if-range)#exit
REDBOX1(config)#prp channel-group 1 supervisionFrameOption vlan-tagged
REDBOX1(config)#prp channel-group 1 supervisionFrameOption vlan-id 10
REDBOX1(config)# spanning-tree bpdufilter enable
REDBOX1(config-if)#spanning-tree portfast edge trunk
```

Add Static Entries to the Node and VDAN Tables

Follow the steps in this section to add a static entry to the node or VDAN table.

Procedure

Step 1 Enter global configuration mode:

configure terminal

Example:

```
switch# configure terminal
switch(config-if)# prp channel-group 1 nodeTableMacaddress 0000.0000.0001 lan-a
```

Step 2 Specify the MAC address to add to the node table for the channel group and specify whether the node is a DAN or a SAN (attached to either LAN-A or LAN-B):

prp channel-group *prp-channel group* **nodeTableMacaddress** *mac-address* {dan | lan-a | lan-b}

prp-channel group: Value of 1 or 2

mac-address: MAC address of the node

Note
Use the **no** form of the command to remove the entry.

Step 3 Specify the MAC address to add to the VDAN table:

prp channel-group *prp-channel group* **vdanTableMacaddress** *mac-address*

prp-channel group: Value of 1 or 2

mac-address: MAC address of the node or VDAN

Note
Use the **no** form of the command to remove the entry.

Clearing All Node Table and VDAN Table Dynamic Entries

Procedure

Step 1 Clear all dynamic entries in the node table by entering the following command:

clear prp node-table [**channel-group** *group*]

Step 2 Clear all dynamic entries in the VDAN table by entering the following command:
clear prp vdan-table [**channel-group** *group*]

If you do not specify a channel group, the dynamic entries are cleared for all PRP channel groups.

Note

The **clear prp node-table** and **clear prp vdan-table** commands clear only dynamic entries. To clear static entries, use the **no** form of the **nodeTableMacaddress** or **vdanTableMacaddress** command shown in Add Static Entries to the Node and VDAN Tables, on page 2550.

Disabling the PRP Channel and Group

Procedure

Step 1 Enter global configuration mode:

configure terminal

Step 2 Disable the PRP channel:

no interface prp-channel *prp-channel-number*

prp-channel number: Value of 1 or 2

Step 3 Exit interface mode:

exit

Errors and Warnings as Syslog Messages

You can configure switches so that errors and warnings become syslogs. Doing so enables you to turn the syslogs into Simple Network Management Protocol (SNMP) traps for proper alerting and maintenance.

The following errors and warnings can be configured to become syslogs:

- Wrong LAN ID A

  The number of frames with a wrong LAN identifier received on port A.

- Wrong LAN ID B

  The number of frames with a wrong LAN identifier received on port B.

- Warning LAN A

  There is a potential problem with the PRP ports for LAN A. (Packet loss condition/Wrong LAN packet counter incremented)

- Warning LAN B

  There is a potential problem with the PRP ports for LAN B. (Packet loss condition/Wrong LAN packet counter incremented)

- Oversize packet A

- Oversize packet B

The parameters in the procedure list are captured from the output of the CLI command **sh prp statistics ingressPacketStatistics**.

You use CLI commands to configure the interval that syslogs are generated, from 60 seconds to 84,400 seconds. The default is 300 seconds. See the section Configure the PRP Logging Interval, on page 2553 in this guide for more information.

Configure the PRP Logging Interval

Complete the following steps to configure a logging interval for the creation of PRP syslogs from errors and warnings. The default is 300 seconds; however, you can choose a value from 60 seconds to 84,400 seconds.

Procedure

At the configuration prompt, enter the following command: **prp logging-interval** ***interval_in_seconds***

To choose the default interval of 300 seconds, do not enter a value. Enter one only to specify a logging inteval other than the 300-second default.

Example:

```
cl_2011#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
cl_2011(config)#prp logging-interval 120
```

The switch generates syslogs from the PRP errors and warnings listed in the section Errors and Warnings as Syslog Messages , on page 2552.

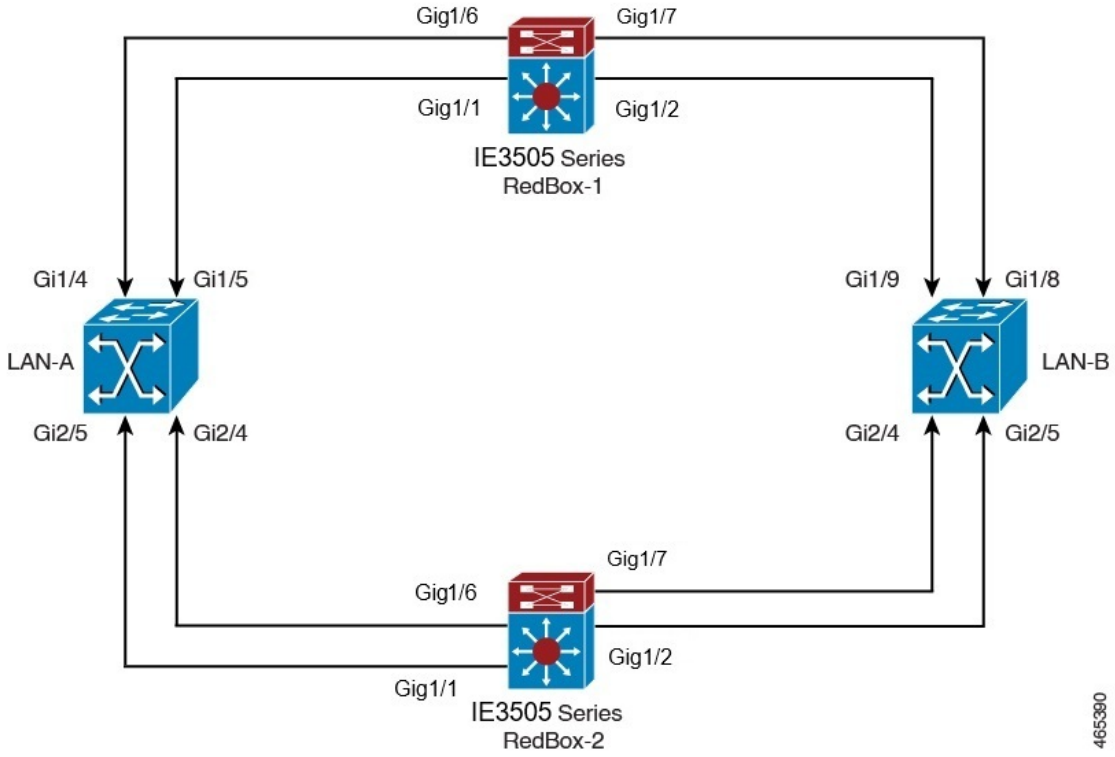
Example

The following text shows sample output resulting from the configuring the logging interval.

```
*Sep 28 13:18:27.623: %PRP_WRONG_LAN-5-WRONG_LAN: PRP channel 2, LAN A is connected to LAN
 B on its peer
*Sep 28 13:18:27.623: %PRP_WRONG_LAN-5-WRONG_LAN: PRP channel 2, LAN B is connected to LAN
 A on its peer
*Sep 28 13:18:27.623: %PRP_WARN_LAN-5-WARN_LAN: PRP channel 2, PRP LAN warning is set on
LAN B
*Sep 28 13:18:27.623: %PRP_OVERSIZE_PKT-5-OVERSIZE_LAN: PRP channel 2, PRP oversize packet
 warning is set on LAN A
```

Configuration Examples

The following diagram shows a network configuration in which the switches might operate. The commands in this example highlight the configuration of features and switches to support that configuration.



In this example, the configuration establishes two LANs, LAN-A and LAN-B, and two PRP channels. Within the topology, a Switch is identified as RedBox-1 and another one is identified as RedBox-2.

Following is the configuration for LAN-A:

```
diagnostic bootup level minimal
!
!
!
spanning-tree mode rapid-pvst
spanning-tree extend system-id
memory free low-watermark processor 88589
!
!
alarm-profile defaultPort
 alarm not-operating
 syslog not-operating
 notifies not-operating
!
!
!
transceiver type all
 monitoring
vlan internal allocation policy ascending
!
!
!
!
!
!
!
!
```

```
!
!
!
!
!
!
interface GigabitEthernet1/1
 shutdown
!
interface GigabitEthernet1/2
 shutdown
!
interface GigabitEthernet1/4
 switchport access vlan 25
 switchport mode access
!
interface GigabitEthernet1/5
 switchport access vlan 35
 switchport mode access
!
interface GigabitEthernet1/6
 shutdown
!
interface GigabitEthernet1/7
 shutdown
!
interface GigabitEthernet1/8
 shutdown
!
interface GigabitEthernet1/9
 shutdown
!
interface GigabitEthernet1/10
 shutdown
!
interface GigabitEthernet1/1
!
interface GigabitEthernet2/1
 shutdown
!
interface GigabitEthernet2/2
 shutdown
!
interface GigabitEthernet2/3
 shutdown
!
interface GigabitEthernet2/4
 switchport access vlan 35
 switchport mode access
!
interface GigabitEthernet2/5
 switchport access vlan 25
 switchport mode access
!
interface GigabitEthernet2/6
 shutdown
!
interface GigabitEthernet2/7
 shutdown
!
interface GigabitEthernet2/8
 shutdown
!
interface Vlan1
```

```
 no ip address
 shutdown
!
interface Vlan35
 no ip address
!
interface Vlan25
 no ip address
```

The configuration for LAN-B is shown below:

```
diagnostic bootup level minimal
!
!
!
spanning-tree mode rapid-pvst
spanning-tree extend system-id
memory free low-watermark processor 88589
!
!
alarm-profile defaultPort
 alarm not-operating
 syslog not-operating
 notifies not-operating
!
!
!
transceiver type all
 monitoring
vlan internal allocation policy ascending
!
!
!
!
!
!
!
!
!
!
!
!
!
interface GigabitEthernet1/1
 shutdown
!
interface GigabitEthernet1/2
 shutdown
!
interface GigabitEthernet1/3
 shutdown
!
interface GigabitEthernet1/4
 shutdown
!
interface GigabitEthernet1/5
 shutdown
!
interface GigabitEthernet1/6
 shutdown
!
interface GigabitEthernet1/7
 shutdown
```

```
!
interface GigabitEthernet1/8
 switchport access vlan 25
 switchport mode access
!
interface GigabitEthernet1/9
 switchport access vlan 35
 switchport mode access
!
interface GigabitEthernet1/10
 shutdown
!
interface AppGigabitEthernet1/1
!
interface GigabitEthernet2/1
 shutdown
!
interface GigabitEthernet2/2
 shutdown
!
interface GigabitEthernet2/3
 shutdown
!
interface GigabitEthernet2/4
 switchport access vlan 35
 switchport mode access
!
interface GigabitEthernet2/5
 switchport access vlan 25
 switchport mode access
!
interface GigabitEthernet2/6
 shutdown
!
interface GigabitEthernet2/7
 shutdown
!
interface GigabitEthernet2/8
 shutdown
!
interface Vlan1
 no ip address
 shutdown
!
interface Vlan35
 no ip address
!
interface Vlan25
 no ip address
```

Following is the configuration for RedBox-1:

```
!
!
spanning-tree mode rapid-pvst
no spanning-tree etherchannel guard misconfig
spanning-tree extend system-id
memory free low-watermark processor 88589
!
!
alarm-profile defaultPort
 alarm not-operating
 syslog not-operating
 notifies not-operating
!
```

```
prp channel-group 1 supervisionFrameOption vlan-id 35
prp channel-group 1 supervisionFrameTime 25000
prp channel-group 1 supervisionFrameLifeCheckInterval 8500
prp channel-group 1 supervisionFrameRedboxMacaddress 34c0.f9e5.59ba
prp channel-group 2 supervisionFrameOption vlan-id 25
prp channel-group 2 supervisionFrameTime 9834
prp channel-group 2 supervisionFrameLifeCheckInterval 12345
prp channel-group 2 passRCT!
!
transceiver type all
 monitoring
vlan internal allocation policy ascending
!
!
!
!
!
!
!
!
!
!
!
!
!
interface PRP-channel1
 switchport access vlan 35
 switchport mode access
 spanning-tree bpdufilter enable
!
interface PRP-channel2
 switchport access vlan 25
 switchport mode access
 spanning-tree bpdufilter enable
!
interface GigabitEthernet1/1
 switchport access vlan 35
 switchport mode access
 no ptp enable
 udld port disable
 no keepalive
 prp-channel-group 1
 spanning-tree bpdufilter enable
!
interface GigabitEthernet1/2
 switchport access vlan 35
 switchport mode access
 no ptp enable
 udld port disable
 no keepalive
 prp-channel-group 1
!
interface GigabitEthernet1/6
 switchport access vlan 25
 switchport mode access
 no ptp enable
 no keepalive
 prp-channel-group 2
 spanning-tree bpdufilter enable
!
interface GigabitEthernet1/7
 switchport access vlan 25
 switchport mode access
```

```
 no ptp enable
 no keepalive
 prp-channel-group 2
 spanning-tree bpdufilter enable

!
interface Vlan1
 no ip address
 shutdown
!
interface Vlan35
 ip address 192.0.2.14 255.255.255.0
!
interface Vlan25
 ip address 192.0.2.15 255.255.255.0
!
interface Vlan100
 ip address 192.0.2.16 255.255.255.0
!
ip http server
ip http authentication local
ip http secure-server
ip forward-protocol nd
!
ip tftp source-interface Vlan100
ip tftp blocksize 8192
!
```

Following is the configuration for RedBox-2:

```
!
spanning-tree mode rapid-pvst
no spanning-tree etherchannel guard misconfig
spanning-tree extend system-id
memory free low-watermark processor 88589
!
!
alarm-profile defaultPort
 alarm not-operating
 syslog not-operating
 notifies not-operating
!
prp channel-group 1 supervisionFrameOption vlan-id 35
prp channel-group 1 supervisionFrameTime 776
prp channel-group 1 supervisionFrameLifeCheckInterval 15000
prp channel-group 1 passRCT
prp channel-group 2 supervisionFrameOption vlan-id 25
prp channel-group 2 supervisionFrameTime 9834
prp channel-group 2 supervisionFrameLifeCheckInterval 12345
prp channel-group 2 passRCT

!
!
!
transceiver type all
 monitoring
vlan internal allocation policy ascending
lldp run
!
!
!
!
!
!
```

```
!
!
!
!
!
!
!
interface PRP-channel1
 switchport access vlan 25
 switchport mode access
 spanning-tree bpdufilter enable
!
interface PRP-channel2
 switchport access vlan 35
 switchport mode access
 spanning-tree bpdufilter enable
!
interface GigabitEthernet1/1
 switchport access vlan 25
 switchport mode access
 no ptp enable
 udld port disable
 no keepalive
 prp-channel-group 1
 spanning-tree bpdufilter enable
!
interface GigabitEthernet1/2
 switchport access vlan 25
 switchport mode access
 no ptp enable
 udld port disable
 no keepalive
 prp-channel-group 1
 spanning-tree bpdufilter enable
!
interface GigabitEthernet1/5
!
interface GigabitEthernet1/6
 description *** PRP 2 channel *****
 switchport access vlan 35
 switchport mode access
 no ptp enable
 no keepalive
 prp-channel-group 2
 spanning-tree bpdufilter enable
!
interface GigabitEthernet1/7
 description *** PRP 2 channel *****
 switchport access vlan 35
 switchport mode access
 no ptp enable
 no keepalive
 prp-channel-group 2
 spanning-tree bpdufilter enable
!
interface AppGigabitEthernet1/1
!
interface Vlan1
 no ip address
 shutdown
!
interface Vlan35
 ip address 192.0.2.14 255.255.255.0
!
```

```
interface Vlan25
 ip address 192.0.2.15 255.255.255.0
!
interface Vlan100
 ip address 192.0.2.16 255.255.255.0
!
ip http server
ip http authentication local
ip http secure-server
ip forward-protocol nd
!
ip tftp source-interface Vlan100
ip tftp blocksize 8192
!
!
!
```

VLAN Tagging Example

The following example shows the configuration of a switch with PRP channel interfaces configured for VLAN tagging of supervision frames.

```
PRP_IE3505# sh running-config
Building configuration...

Current configuration : 8171 bytes
!
! Last configuration change at 05:19:31 PST Mon Mar 22 2025
!
version 17.17
service timestamps debug datetime msec localtime show-timezone
service timestamps log datetime msec localtime show-timezone
service call-home
no platform punt-keepalive disable-kernel-core
no platform punt-keepalive settings
no platform bridge-security all
!
hostname PRP_IE35xx
!
!
no logging console
enable password Cisco123
!
no aaa new-model
clock timezone PST -8 0
rep bpduleak
ptp mode e2etransparent
!
!
!
!
!
!
!
ip dhcp pool webuidhcp
   cip instance 1
!
!
!
login on-success log
!
!
!
crypto pki trustpoint SLA-TrustPoint
```

```
 enrollment pkcs12
 revocation-check crl
!
crypto pki trustpoint TP-self-signed-559094202
 enrollment selfsigned
 subject-name cn=IOS-Self-Signed-Certificate-559094202
 revocation-check none
 rsakeypair TP-self-signed-559094202
!
!
!
diagnostic bootup level minimal
!
!
!
spanning-tree mode rapid-pvst
no spanning-tree etherchannel guard misconfig
spanning-tree extend system-id
memory free low-watermark processor 89983
!
!
alarm-profile defaultPort
 alarm not-operating
 syslog not-operating
 notifies not-operating
!
prp channel-group 1 supervisionFrameOption vlan-tagged
prp channel-group 1 supervisionFrameOption vlan-id 30
prp channel-group 1 supervisionFrameTime 500
prp channel-group 1 supervisionFrameLifeCheckInterval 24907
prp channel-group 1 supervisionFrameRedboxMacaddress ecce.13eb.71a2
prp channel-group 2 supervisionFrameOption vlan-tagged
prp channel-group 2 supervisionFrameOption vlan-id 40
prp channel-group 2 supervisionFrameTime 0
prp channel-group 2 supervisionFrameLifeCheckInterval 0
prp channel-group 2 supervisionFrameRedboxMacaddress f8b7.e2e5.c1f9
!
!
!
transceiver type all
 monitoring
vlan internal allocation policy ascending
lldp run
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
interface PRP-channel1
 switchport mode trunk
 switchport trunk allowed vlan 30,40

 spanning-tree bpdufilter enable
!
interface PRP-channel2
```

```
 switchport mode trunk
 switchport trunk allowed vlan 30,40
 no keepalive
 spanning-tree bpdufilter enable
!
interface GigabitEthernet1/1
 switchport mode trunk
 switchport trunk allowed vlan 30,40
 no ptp enable
 udld port disable
 no keepalive
 prp-channel-group 1
 spanning-tree bpdufilter enable
!
interface GigabitEthernet1/2
 switchport mode trunk
 switchport trunk allowed vlan 30,40
 no ptp enable
 udld port disable
 no keepalive
 prp-channel-group 1
 spanning-tree bpdufilter enable

!
interface GigabitEthernet1/6
 switchport mode trunk
 switchport trunk allowed vlan 30,40
 no ptp enable
 udld port disable
 no keepalive
 prp-channel-group 2
 spanning-tree bpdufilter enable
!
interface GigabitEthernet1/7
 switchport mode trunk
 switchport trunk allowed vlan 30,40
 no ptp enable
 udld port disable
 no keepalive
 prp-channel-group 2
 spanning-tree bpdufilter enable
!
interface Vlan1
 no ip address
 shutdown
!
interface Vlan30
 ip address 192.0.2.17 255.255.255.0
!
interface Vlan40
 ip address 192.0.2.18 255.255.255.0
!
interface Vlan197
 ip address 192.0.2.19 255.255.255.0
!
ip http server
ip http authentication local
ip http secure-server
ip forward-protocol nd
!
ip tftp source-interface Vlan197
ip tftp blocksize 8192
!
!
```

```
!
!
!
!
control-plane
!
!
line con 0
 exec-timeout 0 0
 stopbits 1
line aux 0
line vty 0 4
 login
 transport input ssh
line vty 5 15
 login
 transport input ssh
!
call-home
 ! If contact email address in call-home is configured as sch-smart-licensing@cisco.com
 ! the email address configured in Cisco Smart License Portal will be used as contact email
 address to send SCH notifications.
 contact-email-addr sch-smart-licensing@cisco.com
 profile "CiscoTAC-1"
  active
  destination transport-method http
!
!
!
!
!
!
!
!
!
!
end

PRP_IE35xx#
```

Verify Configuration

This section lists commands that you can use to verify PRP configuration and examples of those commands.

Command	Purpose
show prp channel {**1** | **2** [**detail** | **status** | **summary**] | **detail** | **status** | **summary**}	Displays configuration details for a specified PRP channel.
show prp control {**VdanTableInfo** | **ptpLanOption** | **ptpProfile** | **supervisionFrameLifeCheckInterval** | **supervisionFrameOption** | **supervisionFrameRedboxMacaddress** | **supervisionFrameTime**}	Displays PRP control information, VDAN table, and supervision frame information.
show prp node-table [**channel-group** <**group**> | **detail**]	Displays PRP node table.

Command	Purpose
show prp statistics {**egressPacketStatistics** | **ingressPacketStatistics** | **nodeTableStatistics** | **pauseFrameStatistics** | **ptpPacketStatistics**}	Displays statistics for PRP components.
show prp vdan-table [**channel-group** <**group**> | **detail**]	Displays PRP VDAN table.
show interface prp-channel {**1** | **2**}	Displays information about PRP member interfaces.

Note The **show interface G1/1** or **show interface G1/2** command should not be used to read PRP statistics if these interfaces are PRP channel members because the counter information can be misleading. Use the **show interface prp-channel** [**1** | **2**] command instead.

The following example shows the output for **show prp channel** when one of the interfaces in the PRP channel is down.

```
show prp channel 1 detail
PRP-channel: PR1
------------
 Layer type = L2
 Ports: 2       Maxports = 2
 Port state = prp-channel is Inuse
 Protocol = Enabled
Ports in the group:
  1) Port: Gi1/1
   Logical slot/port = 1/1      Port state = Inuse
        Protocol = Enabled
  2) Port: Gi1/2
   Logical slot/port = 1/2      Port state = Inuse
        Protocol = Enabled
```

The following example shows how to display the PRP node table and PRP VDAN table.

```
Switch#show prp node-table
PRP Channel 1 Node Table
==================================
   Mac Address   Type  Dyn   TTL
---------------- ----- --- -------
 B0AA.7786.6781  lan-a  Y     59
 F454.3317.DC91  dan    Y     60
==================================
Channel 1 Total Entries: 2
Switch#show prp vdan-table
PRP Channel 1 VDAN Table
============================
   Mac Address   Dyn   TTL
---------------- --- -------
 F44E.05B4.9C81   Y     60
============================
Channel 1 Total Entries: 1
```

The following example shows output for the **show prp control supervisionFrameOption** command with and without VLAN tagging added to the PRP channel. A `VLAN value` field of 1 means that VLAN tagging is enabled, and a value of 0 means that VLAN tagging is disabled.

```
REDBOX1#show prp control supervisionFrameoption
 PRP channel-group 1 Super Frame Option
  COS value is 7
  CFI value is 0
  VLAN value is 1
  MacDA value is 200
  VLAN id value is 30
PRP channel-group 2 Super Frame Option
  COS value is 0
  CFI value is 0
  VLAN value is 0
  MacDA value is 0
  VLAN id value is 0

REDBOX1#
```

The following example shows the command to determine if the switch has been configured so that errors and warnings to become syslogs:

```
switch #sh prp control logging-interval
 PRP syslog logging interval is not configured
```

The following example shows the command for configuring the logging interval to the default, 300 seconds.

```
switch #conf t
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)#prp logging-interval
switch(config)#do sh prp control logging-interval
 PRP syslog logging interval is 300 in seconds
```

The following example shows the command for configuring the logging interval to 600 seconds.

```
switch(config)#prp logging-interval 600
 PRP syslog logging interval is 600 in seconds

switch(config)#
```

CHAPTER 173

Resilient Ethernet Protocol

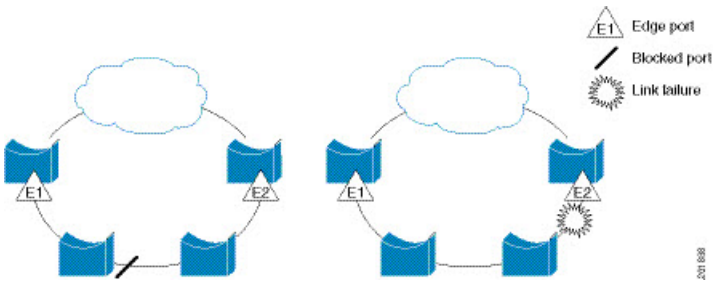
Resilient Ethernet Protocol

Resilient Ethernet Protocol (REP) is a Cisco proprietary protocol that provides an alternative to Spanning Tree Protocol (STP) to control network loops, handle link failures, and improve convergence time. REP controls a group of ports that are connected in a segment, ensures that the segment does not create any bridging loops, and responds to link failures within the segment. REP provides a basis for constructing more complex networks and supports VLAN load balancing.

REP segment is a chain of ports that are connected to each other and configured with a segment ID. Each segment consists of standard (nonedge) segment ports and two user-configured edge ports. A switch can have no more than two ports that belong to the same segment, and each segment port can have only one external neighbor. A segment can go through a shared medium, but on any link, only two ports can belong to the same segment. **REP is supported only on Trunk ports**.

The following figure shows an example of a segment consisting of six ports spread across four switches. Ports E1 and E2 are configured as edge ports. When all ports are operational (as in the segment on the left), a single port is blocked, shown by the diagonal line. This blocked port is also known as the Alternate port (ALT port). When there is a failure in the network, the blocked port returns to the forwarding state to minimize network disruption.

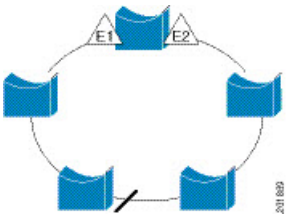
Figure 169: REP Open Segment



The segment shown in the preceding figure is an open segment; there is no connectivity between the two edge ports. The REP segment cannot cause a bridging loop, and you can safely connect the segment edges to any network. All hosts connected to switches inside the segment have two possible connections to the rest of the network through the edge ports, but only one connection is accessible at any time. If a failure occurs on any segment or on any port on a REP segment, REP unblocks the ALT port to ensure that connectivity is available through the other gateway.

The segment in the following figure is a closed segment, also known as Ring Segment, with both edge ports located on the same router. With this configuration, you can create a redundant connection between any two routers in the segment.

Figure 170: REP Ring Segment



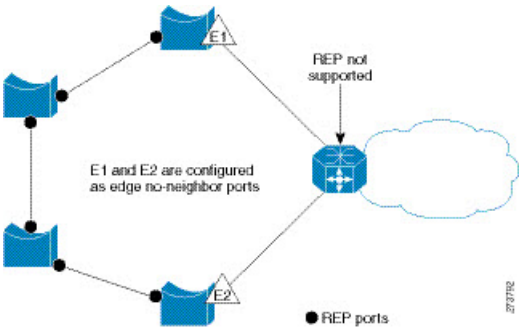
REP segments have the following characteristics:

- If all ports in a segment are operational, one port (referred to as the ALT port) is in the blocked state for each VLAN. If VLAN load balancing is configured, two ALT ports in the segment control the blocked state of VLANs.
- If a port is not operational, and causes a link failure, all ports forward traffic on all VLANs to ensure connectivity.
- In case of a link failure, alternate ports are unblocked as quickly as possible. When the failed link is restored, a logically blocked port per VLAN is selected with minimal disruption to the network.

You can construct almost any type of network that is based on REP segments.

In access ring topologies, the neighboring switch might not support REP as shown in the following figure. In this case, you can configure the non-REP facing ports (E1 and E2) as edge no-neighbor ports. The edge no-neighbor port can be configured to send an STP topology change notice (TCN) towards the aggregation switch.

Figure 171: Edge No-Neighbor Ports



REP has these limitations:

- You must configure each segment port; an incorrect configuration can cause forwarding loops in the networks.
- REP can manage only a single failed port within the segment; multiple port failures within the REP segment cause loss of network connectivity.
- You should configure REP only in networks with redundancy. Configuring REP in a network without redundancy causes loss of connectivity.

Link Integrity

REP does not use an end-to-end polling function between edge ports to verify link integrity. It implements local link failure detection. The REP Link Status Layer (LSL) detects its REP-aware neighbor and establishes connectivity within the segment. All the VLANs are blocked on an interface until the neighbor is detected. After the neighbor is identified, REP determines which neighbor port should become the alternate port and which ports should forward traffic.

Each port in a segment has a unique port ID. The port ID format is similar to that used by the spanning tree algorithm: a port number (unique on the bridge) associated to a MAC address (unique in the network). When a segment port is coming up, its LSL starts sending packets that include the segment ID and the port ID. The port is declared as operational after it performs a three-way handshake with a neighbor in the same segment.

A segment port does not become operational if:

- No neighbor has the same segment ID.
- More than one neighbor has the same segment ID.
- A neighbor does not acknowledge a local port as a peer.

Each port creates an adjacency with its immediate neighbor. After the neighbor adjacencies are created, the ports negotiate with each other to determine the blocked port for the segment, which will function as the alternate port. All the other ports become unblocked. By default, REP packets are sent to a bridge protocol data unit-class MAC address. The packets can also be sent to a Cisco multicast address, which is used only to send blocked port advertisement (BPA) messages when there is a failure in the segment. The packets are dropped by the devices not running REP.

Fast Convergence

REP runs on a physical link basis and not on a per-VLAN basis. Only one hello message is required for all the VLANs, and this reduces the load on the protocol. We recommend that you create VLANs consistently on all the switches in a given segment and configure the same allowed VLANs on the REP trunk ports. To avoid the delay introduced by relaying messages in software, REP also allows some packets to be flooded to a regular multicast address. These messages operate at the hardware flood layer (HFL) and are flooded to the entire network, not just the REP segment. Switches that do not belong to the segment treat them as data traffic. You can control flooding of these messages by configuring an administrative VLAN for the entire domain or for a particular segment.

VLAN Load Balancing

One edge port in the REP segment acts as the primary edge port; and another as the secondary edge port. It is the primary edge port that always participates in VLAN load balancing in the segment. REP VLAN balancing

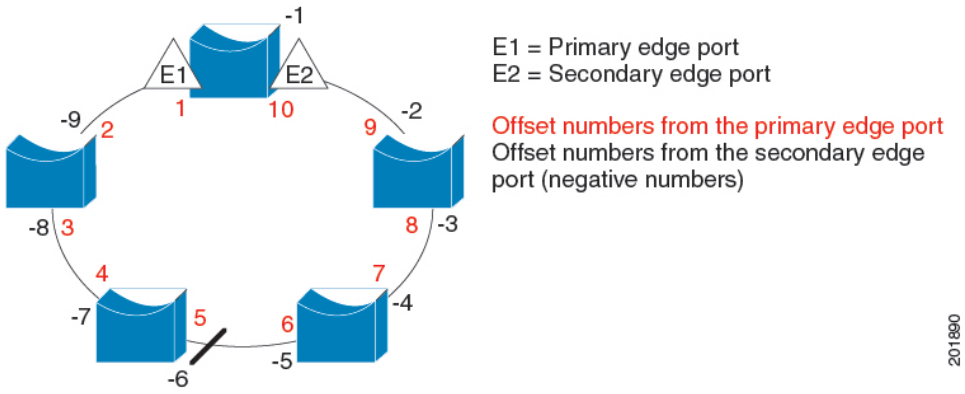
is achieved by blocking some VLANs at a configured alternate port and all the other VLANs at the primary edge port. When you configure VLAN load balancing, you can specify the alternate port in one of three ways:

- By entering the port ID of the interface. To identify the port ID of a port in the segment, enter the **show interface rep detail** interface configuration command for the port.
- By entering the **preferred** keyword to select the port that you previously configured as the preferred alternate port with the **rep segment** *segment-id* **preferred** interface configuration command.
- By entering the neighbor offset number of a port in the segment, which identifies the downstream neighbor port of an edge port. The neighbor offset number range is –256 to +256; a value of 0 is invalid. The primary edge port has an offset number of 1; positive numbers above 1 identify downstream neighbors of the primary edge port. Negative numbers indicate the secondary edge port (offset number -1) and its downstream neighbors.

Note Configure offset numbers on the primary edge port by identifying a port's downstream position from the primary (or secondary) edge port. Never enter an offset value of 1 because that is the offset number of the primary edge port.

The following figure shows neighbor offset numbers for a segment, where E1 is the primary edge port and E2 is the secondary edge port. The red numbers inside the ring are numbers offset from the primary edge port; the black numbers outside of the ring show the offset numbers from the secondary edge port. Note that you can identify all the ports (except the primary edge port) by either a positive offset number (downstream position from the primary edge port) or a negative offset number (downstream position from the secondary edge port). If E2 became the primary edge port, its offset number would then be 1 and E1 would be -1.

Figure 172: Neighbor Offset Numbers in a Segment



When the REP segment is complete, all the VLANs are blocked. When you configure VLAN load balancing, you must also configure triggers in one of two ways:

- Manually trigger VLAN load balancing at any time by entering the **rep preempt segment** *segment-id* privileged EXEC command on the switch that has the primary edge port.
- Configure a preempt delay time by entering the **rep preempt delay** *seconds* interface configuration command. After a link failure and recovery, VLAN load balancing begins after the configured preemption time period elapses. Note that the delay timer restarts if another port fails before the time has elapsed.

Note When VLAN load balancing is configured, it does not start working until triggered by either manual intervention or a link failure and recovery.

When VLAN load balancing is triggered, the primary edge port sends out a message to alert all the interfaces in the segment about the preemption. When the secondary port receives the message, the message is sent to the network to notify the alternate port to block the set of VLANs specified in the message and to notify the primary edge port to block the remaining VLANs.

You can also configure a particular port in the segment to block all the VLANs. Only the primary edge port initiates VLAN load balancing, which is not possible if the segment is not terminated by an edge port on each end. The primary edge port determines the local VLAN load-balancing configuration.

Reconfigure the primary edge port to reconfigure load balancing. When you change the load-balancing configuration, the primary edge port waits for the **rep preempt segment** command or for the configured preempt delay period after a port failure and recovery, before executing the new configuration. If you change an edge port to a regular segment port, the existing VLAN load-balancing status does not change. Configuring a new edge port might cause a new topology configuration.

Spanning Tree Interaction

REP does not interact with STP, but it can coexist. A port that belongs to a segment is removed from spanning tree control and STP BPDUs are not accepted or sent from segment ports. Therefore, STP cannot run on a REP segment.

To migrate from an STP ring configuration to REP segment configuration, begin by shutdown the interface and proceed with configuring a single port in the ring as part of the segment and continue by configuring contiguous ports to minimize the number of segments.

Each segment always contains a blocked port, so multiple segments means multiple blocked ports and a potential loss of connectivity. When the segment has been configured in both directions up to the location of the edge ports, you then configure the edge ports. After the configuration, enable or unshut the ports.

REP Ports

REP segments consist of Failed, Open, or Alternate ports:

- A port configured as a regular segment port starts as a failed port.
- After the neighbor adjacencies are determined, the port transitions to alternate port state, blocking all the VLANs on the interface. Blocked-port negotiations occur, and when the segment settles, one blocked port remains in the alternate role and all the other ports become open ports.
- When link flap is triggered, only the link that is shut moves to Failed state. When the Alternate port receives the failure notification, it changes to the Open state, forwarding all the VLANs.

A regular segment port converted to an edge port, or an edge port converted to a regular segment port, does not always result in a topology change. If you convert an edge port into a regular segment port, VLAN load balancing is not implemented unless it has been configured. **For VLAN load balancing, you must configure two edge ports in the segment.**

A segment port that is reconfigured as a spanning tree port restarts according to the spanning tree configuration. By default, this is a designated blocking port. If PortFast is configured or if STP is disabled, the port goes into the forwarding state.

Resilient Ethernet Protocol Fast

The Resilient Ethernet Protocol (REP) Fast feature allows faster link failure detection and convergence on the switch copper Gigabit Ethernet (GE) ports.

On Fiber GE ports, link down detection time is also 10 ms, but on GE copper interfaces, the link drop detection and recovery times are between 750 ms and 350 ms. As a result, link loss and recovery can be detected a lot more quickly on GE fiber interfaces than on corresponding copper interfaces. This in turn means that the convergence time for REP is a lot higher when using GE copper interfaces.

To improve link down detection time, a beacon mechanism is implemented to trigger faster link failure detection when a REP interface is configured for REP Fast mode. The switch has two timers for each REP interface. The first timer is triggered every 3 ms to transmit the beacon frame to the neighbor node. After successful transmission and reception of the frame, both the timers are reset. If the packet is not received after the transmission, then the second timer is triggered to check the reception within 10 ms. If the packet is not received, upon the timer expiry, a link down message is sent to the switch.

REP Fast works on an individual link basis. It does not impact the REP Protocol. REP Fast requires both ends of the link to support REP Fast to work. REP Fast can be used on any interface link pair that is configured for REP, but it was created to solve an issue on Gigabit copper links. REP Fast speeds up detection of the link failure on Gigabit copper interfaces.

A switch can have a combination of REP rings and REP Fast rings, with each configured as a separate segment. REP Fast enablement does not impact REP ring size since it operates only on the pair of interfaces that are configured for it. Because REP Fast has to generate Beacon frames, only six interfaces on a single REP node can be configured for REP Fast at a time.

Note It is recommended to not have a mix of REP and REP Fast on a single ring.

If the neighbor acknowledges and is configured for REP Fast mode, convergence occurs within 50 ms. If a neighbor switch does not support the REP Fast feature, normal REP mode must be used for link up/down detection. In this case, you must disable fast mode on both ends of the link.

For information about configuring REP Fast, see Configure REP Fast in this guide.

Configure REP Fast

Follow these steps to configure REP Fast:

Before you begin

Enable REP on the switch and configure the REP topology as described in Configuring REP.

Procedure

Step 1 Enter global configuration mode:

configure terminal

Step 2 Specify the interface and enter interface configuration mode:

interface *interface-id*

Step 3 Enable REP Fast:

REP fastmode

Step 4 Return to privileged exec mode:

end

Example

```
gigabitethernet1/1
switch(config-if)#rep seg
switch(config-if)#rep segment ?
<1-1024> Between 1 and 1024

switch(config-if)#rep segment 10
switch(config-if)#rep fastmode
switch(config)#int <interface number>
switch(config-if)#
switch(config-if)#rep ?
  fastmode       REP fastmode
switch (config-if)#rep fastmode ?
  <cr>  <cr>

switch#sh run int <interface number>
Building configuration...

Current configuration : 89 bytes
!
interface <interface number>
 switchport mode trunk
 rep segment <segment id>
 rep fastmode
end
switch#

switch#sh run int <interface number>
Building configuration...

Current configuration : 89 bytes
!
interface <interface number>
 switchport mode trunk
 rep segment <segment id>
 rep fastmode
end
```

REP Zero Touch Provisioning

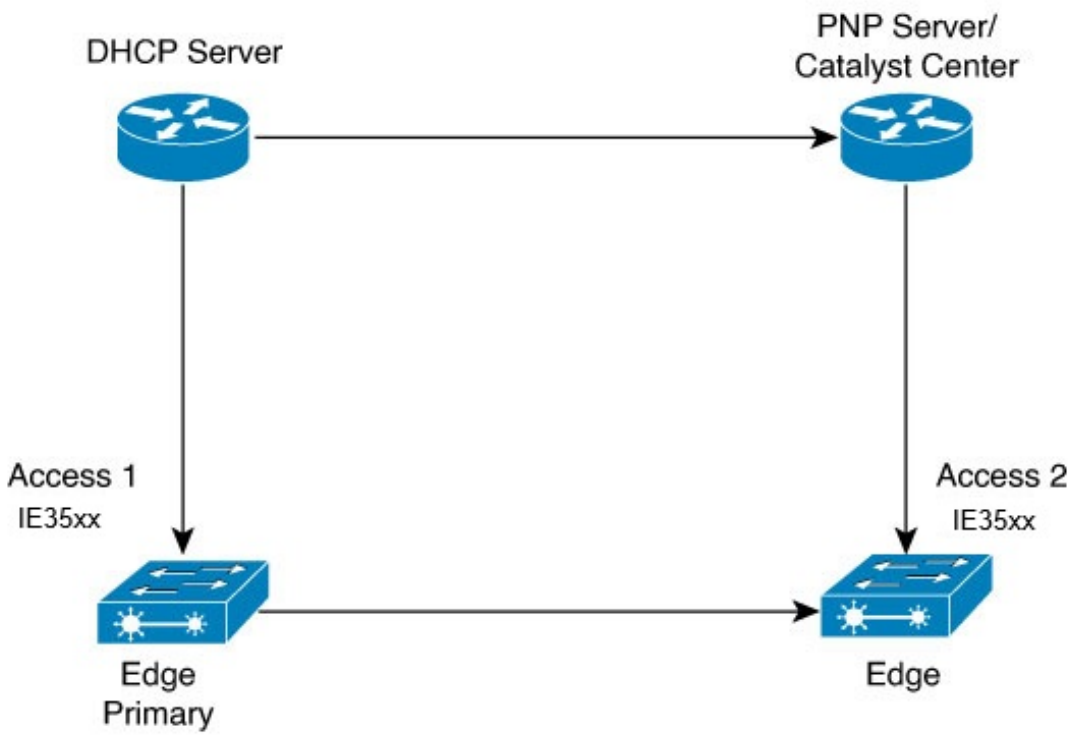
Before a network device such as a router or a switch is deployed online and fully functional, a fair amount of manual configuration is required. Zero Touch Provisioning (ZTP) technologies automate these processes, bringing up network devices into a functional state with minimal to no manual configuration. The Cisco Network Plug and Play (PnP) and Autoinstall Day Zero solutions provide a simple, secure, unified, and integrated offering for enterprise and industrial network customers to ease device rollouts for provisioning updates to an existing network. However, PnP does not support Resilient Ethernet Protocol (REP) due to the way REP is designed. Prior to the REP ZTP feature, REP ring provisioning for Day Zero required manual intervention. The REP ZTP feature introduces a new type-length-value (TLV) extension into the REP LSL packets to support configuring REP rings with zero-touch technologies.

REP and Day Zero

In a typical switch deployment using ZTP, the switch, with no startup configuration in the NVRAM, triggers the Cisco Open Plug-n-Play (PnP) agent to initiate a DHCP discovery process. This process acquires the IP configuration required for the switch from the DHCP server. The DHCP server can be configured to insert additional information in a DHCP message using vendor specific option 43. After the DHCP server receives a DHCP DISCOVER message with option 60 and the string "cisco pnp" from the switch, the DHCP server sends the IP address or hostname of the PnP server to the requesting switch. When the switch receives the DHCP response, the PnP agent extracts the option 43 from the response to get the IP address or the hostname of the PnP server. The PnP agent on the switch then uses this IP address or hostname to communicate with the PnP server. Finally, the PnP server downloads the required Day Zero configuration to the switch to complete the provisioning.

The example shown in the following diagrams illustrates REP ring provisioning on Day Zero, prior to the introduction of REP ZTP.

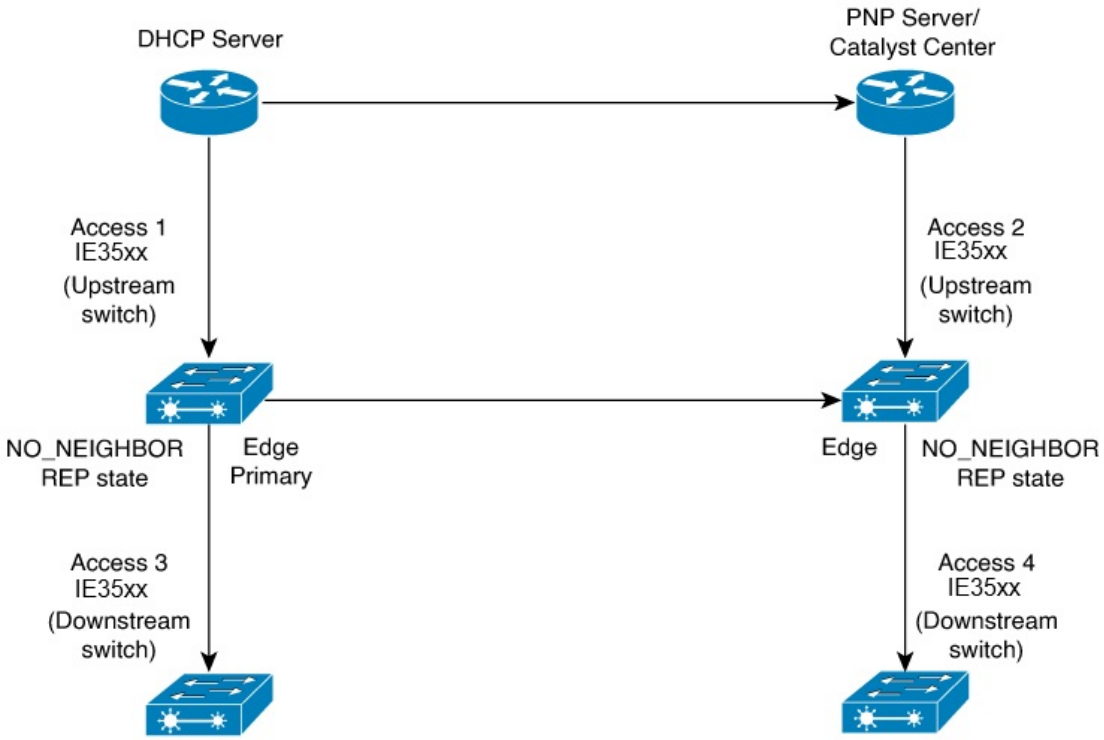
Figure 173: Adding Edge Nodes to the REP Ring



Note The DHCP Server and the PnP Server/Cisco Catalyst Center are not part of the REP ring.

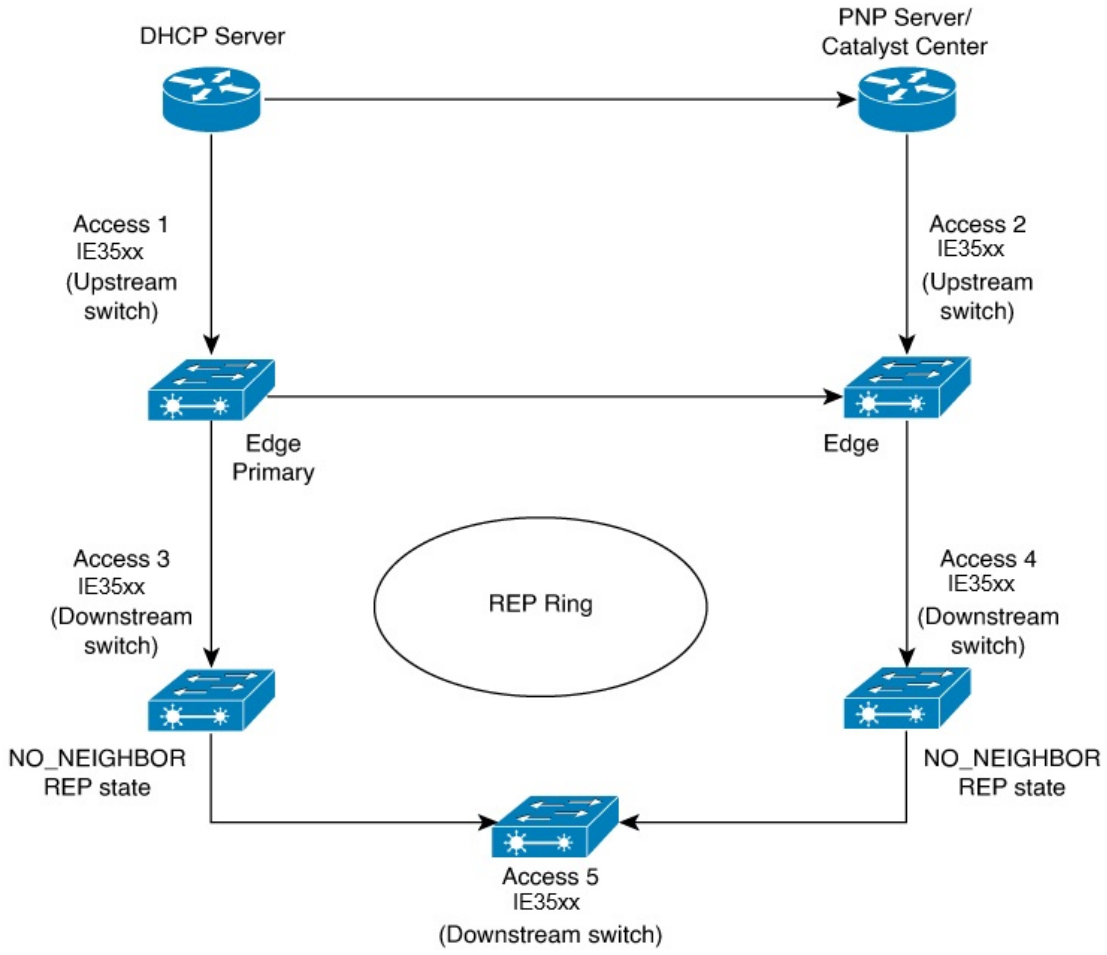
The first set of nodes to be provisioned are Access 1 and Access 2 in the diagram. These are the 2 edge nodes of the REP ring. Note that PnP has configured the downlink port as primary edge on Access 1 and secondary edge on Access 2.

Figure 174: Adding Downstream Nodes



When either Access 3 or Access 4 are powered on, the REP edge primary port starts the REP protocol negotiation and discovers that the neighbor port is not a REP enabled port. (Recall that the switch will be added to the REP ring only after PnP provisioning, for which it needs to first contact the DHCP server as explained earlier.) When an upstream switch port has REP configured and a downstream switch is getting on-boarded with PnP, the REP port goes into the NO_NEIGHBOR state because it is not able to discover its REP peer. In the NO_NEIGHBOR state, REP blocks all the VLANs on that port. This means that the DHCP discovery message from the new switch on the PnP startup VLAN is dropped by the upstream switch because its REP state is NO_NEIGHBOR. The same sequence of blocked ports continues for all new switches added to the REP ring (see Access 5 in figure below).

Figure 175: NO_NEIGHBOR REP State



REP ZTP Overview

The REP ZTP enhancements require that both the upstream and the downstream switches support the feature. When the new downstream switch is powered on, it initiates PNP/autoinstall. The upstream switch's interface is configured for REP and blocks the interface to the downstream switch because the downstream switch is not REP by default (the upstream switch is in REP_NO_NEIGHBOR state).

Even though the interface on the upstream switch is blocked, it will transmit REP LSL packets to the downstream switch. This is normal. With the enhancement of the REP ZTP feature, the downstream switch will start transmitting REP LSL packets with a new TLV to inform the upstream switch that its neighbor is attempting PNP provisioning.

When the upstream switch reads this REP LSL with the new TLV, it will unblock the interface for the PNP startup VLAN only. All other VLANs for which the upstream interface is a member continue to be blocked. Because the upstream switch is forwarding packets on the PNP startup VLAN for this interface, the downstream switch can complete the PNP process.

The intent of this feature is to allow new switches to join a REP ring with no manual intervention. The interface on the upstream switch keeps the startup VLAN unblocked until the downstream switch has received its

configuration and has configured its own interface for REP. If there's a failure in the PNP proccess, the interface on the upstream switch reverts to blocking on the PNP startup VLAN. If the configuration received by the downstream switch does configure the interface for REP, the upstream switch reverts to blocking the PNP startup VLAN.

The downstream behavior to transmit the REP LSL with new TLV to request the PnP startup VLAN be unblocked is the default behavior for switches with no startup configuration. For security purposes, the upstream switch must have the interface to the downstream switch explicitly enabled to put the PnP startup VLAN into unblocked state. The interface level command is **rep ztp-enable**. See Configuring REP ZTP , on page 2587.

Note The upstream switch can be part of multiple REP rings and thereby connected to multiple downstream neighbours. The PnP startup VLAN is unblocked only on the interfaces to which the downstream switch is connected.

Configuring Resilient Ethernet Protocol

A segment is a collection of ports that are connected to one another in a chain and configured with a segment ID. To configure REP segments, configure the REP administrative VLAN (or use the default VLAN 1) and then add the ports to the segment, using interface configuration mode. You should configure two edge ports in a segment, with one of them being the primary edge port and the other the secondary edge port by default. A segment should have only one primary edge port. If you configure two ports in a segment as primary edge ports, for example, ports on different switches, the REP selects one of them to serve as the segment's primary edge port. If necessary, you can configure the location to which segment topology change notices (STCNs) and VLAN load balancing are to be sent.

Default REP Configuration

- REP is disabled on all the interfaces. When enabled, the interface is a regular segment port unless it is configured as an edge port.
- When REP is enabled, the task of sending segment topology change notices (STCNs) is disabled, all the VLANs are blocked, and the administrative VLAN is VLAN 1.
- When VLAN load balancing is enabled, the default is manual preemption with the delay timer disabled. If VLAN load balancing is not configured, the default after manual preemption is to block all the VLANs in the primary edge port.
- REP Fast is disabled by default.
- REP Zero Touch Provisioning is enabled by default at the global level and disabled at the interface level.

REP Configuration Guidelines and Limitations

Follow these guidelines when configuring REP:

- We recommend that you begin by configuring one port and then configure contiguous ports to minimize the number of segments and the number of blocked ports.

- If more than two ports in a segment fail when no external neighbors are configured, one port goes into a forwarding state for the data path to help maintain connectivity during configuration.

  In the **show interfaces rep** command output, the Port Role for this port shows as “Fail Logical Open”; the Port Role for the other failed port shows as “Fail No Ext Neighbor”. When the external neighbors for the failed ports are configured, the ports go through the alternate port state transitions and eventually go to an open state or remain as the alternate port, based on the alternate port selection mechanism.

- REP ports must be Layer 2 IEEE 802.1Q or trunk ports.
- We recommend that you configure all trunk ports in the segment with the same set of allowed VLANs.
- **Be careful when configuring REP through a SSH or Telnet connection. Because REP blocks all VLANs until another REP interface sends a message to unblock it. You might lose connectivity to the router if you enable REP in a SSH or Telnet session that accesses the router through the same interface.**
- If any configuration change is required. Shutdown or disable interface, apply configuration changes, and unshut or enable interface.
- You cannot run REP and STP on the same segment or interface.
- If you connect an STP network to a REP segment, be sure that the connection is at the segment edge. An STP connection that is not at the edge could cause a bridging loop because STP does not run on REP segments. All STP BPDUs are dropped at REP interfaces.
- If REP is enabled on two ports on a switch, both ports must be either regular segment ports or edge ports. REP ports follow these rules:
  - Only two ports on a switch can belong to the same REP segment.
  - If only one port on a switch is configured in a segment, the port should be an edge port.
  - If two ports on a switch belong to the same segment, they must be both edge ports, both regular segment ports, or one regular port and one edge no-neighbor port. An edge port and regular segment port on a switch cannot belong to the same segment.
  - If two ports on a switch belong to the same segment and one is configured as an edge port and one as a regular segment port (a misconfiguration), the edge port is treated as a regular segment port.
- REP interfaces come up in a blocked state and remain in a blocked state until they are safe to be unblocked. You must be aware of this status to avoid sudden connection losses.
- REP sends all LSL PDUs in untagged frames on the native VLAN. The BPA message sent to the Cisco multicast address is sent on the administration VLAN, which is VLAN 1 by default.
- You can configure how long a REP interface remains up without receiving a hello from a neighbor. You can use the **rep lsl-age-timer** value interface configuration command to set the time from 120 ms to 10000 ms. The LSL hello timer is then set to the age-timer value divided by three. In normal operation, three LSL hellos are sent before the age timer on the peer switch expires and checks for hello messages. Only use **rep lsl-age-timer** for non-REP Fast copper Gigabit interfaces. All other interfaces do not benefit from **rep lsl-age-timer**.
  - EtherChannel port channel interfaces do not support LSL age-timer values less than 1000 ms. If you try to configure a value less than 1000 ms on a port channel, you receive an error message and the command is rejected.

- **lsl-age-timer** is intended to be used when normal link down detection will be too slow for convergence time.

  FastEthernet and fiber connections do not need **lsl-age-timer**. Gigabit copper can use REP Fast instead of **lsl-age-timer**.

- REP ports cannot be configured as one of the following port types:
  - Switched Port Analyzer (SPAN) destination port
  - Tunnel port
  - Access port
- REP is supported on EtherChannels, but not on an individual port that belongs to an EtherChannel.
- Switch supports maximum of 5 REP segments or 3 REP Fast segments with the base module.

Note To increase the number of REP rings, an expansion module can be attached to the base module.

- There is no limit to the size of a REP ring. REP ring sizes greater than 20 nodes may not achieve desired convergence.

Follow these guidelines when configuring REP Fast:

- You must configure REP Fast on both ends of the link in order for the feature to work.
- A REP segment can contain a mix of Gigabit fiber and Gigabit copper. The 50 ms requirement for convergence from a single failure can be achieved if Gigabit copper interfaces have REP Fast.
- Be aware of the following limitations:

Note It is recommended to not have a mix of REP and REP Fast on a single ring.

- A maximum of three REP segments can have REP Fast enabled..
- MAC Sec is supported on both REP and REP Fast.
- REP Fast over EtherChannel is not supported.

REP ZTP Configuration Guidelines

- REP ZTP requires the PnP feature to be present on the switches.
- This transient state change in port forwarding behavior in NO_NEIGHBOR state allows a DHCP request message to reach a DHCP server and unblock PnP provisioning of a new switch. There should not be any impact to the REP state machine after PnP completion.
- The changes in REP behavior during the NO_NEIGHBOR state apply only to REP Zero Touch Provisioning (ZTP). If the PnP feature is not present, normal REP functionality should work as expected.

- REP ZTP is supported on physical and EtherChannel interfaces.
- REP ZTP is supported on both copper (downlink) and fiber (uplink) interfaces.
- REP ZTP is interoperable only with other IE switching products running IOS XE that claim REP ZTP support.

Configure REP Administrative VLAN

To avoid the delay created by link-failure messages, and VLAN-blocking notifications during load balancing, REP floods packets to a regular multicast address at the hardware flood layer (HFL). These messages are flooded to the whole network, and not just the REP segment. You can control the flooding of these messages by configuring an administrative VLAN.

Follow these guidelines when configuring the REP administrative VLAN:

- If you do not configure an administrative VLAN, the default is VLAN 1.
- You can configure one admin VLAN on the switch for all segments.
- The administrative VLAN cannot be the RSPAN VLAN.

To configure the REP administrative VLAN, follow these steps, beginning in privileged EXEC mode:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**rep admin vlan** *vlan-id* **Example:** `Device(config)# rep admin vlan 2`	Specifies the administrative VLAN. The range is from 2 to 4094. To set the admin VLAN to 1, which is the default, enter the **no rep admin vlan** global configuration command.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**show interface** [*interface-id*] **rep detail** **Example:** `Device# show interface gigabitethernet1/1 rep detail`	(Optional) Verifies the configuration on a REP interface.

	Command or Action	Purpose
Step 6	**copy running-config startup config** **Example:** `Device# copy running-config startup config`	(Optional) Saves your entries in the switch startup configuration file.

Configure a REP Interface

To configure REP, enable REP on each segment interface and identify the segment ID. This task is mandatory, and must be done before other REP configurations. You must also configure a primary and secondary edge port on each segment. All the other steps are optional.

Follow these steps to enable and configure REP on an interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/1`	Specifies the interface, and enters interface configuration mode. The interface can be a physical Layer 2 interface or a port channel (logical interface).
Step 4	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Configures the interface as a Layer 2 trunk port.
Step 5	**rep segment** *segment-id* [**edge** [**no-neighbor**] [**primary**]] [**preferred**] **Example:** `Device(config-if)# rep segment 1 edge no-neighbor primary`	Enables REP on the interface and identifies a segment number. The segment ID range is from 1 to 1024. **Note** You must configure two edge ports, including one primary edge port, for each segment. These optional keywords are available: • (Optional) **edge**: Configures the port as an edge port. Each segment has only two edge ports. Entering the keyword **edge**

	Command or Action	Purpose
		without the keyword **primary** configures the port as the secondary edge port. • (Optional) **primary**: Configures the port as the primary edge port, the port on which you can configure VLAN load balancing. • (Optional) **no-neighbor**: Configures a port with no external REP neighbors as an edge port. The port inherits all the properties of an edge port, and you can configure the properties the same way you do for an edge port. **Note** Although each segment can have only one primary edge port, if you configure edge ports on two different switches and enter the keyword **primary** on both the switches, the configuration is valid. However, REP selects only one of these ports as the segment primary edge port. You can identify the primary edge port for a segment by entering the **show rep topology** command in privileged EXEC mode. • (Optional) **preferred**: Indicates that the port is the preferred alternate port or the preferred port for VLAN load balancing. **Note** Configuring a port as preferred does not guarantee that it becomes the alternate port; it merely gives the port a slight edge over equal contenders. The alternate port is usually a previously failed port.
Step 6	**rep stcn** {**interface** *interface id* | **segment** *id-list* | **stp**} **Example:** `Device(config-if)# rep stcn segment 25-50`	(Optional) Configures the edge port to send segment topology change notices (STCNs). • **interface** *interface-id*: Designates a physical interface or port channel to receive STCNs. • **segment** *id-list*: Identifies one or more segments to receive STCNs. The range is from 1 to 1024. • **stp**: Sends STCNs to STP networks. **Note**

	Command or Action	Purpose
		Spanning Tree (MST) mode is required on edge no-neighbor nodes when **rep stcn stp** command is configured for sending STCNs to STP networks. **Note** The incorrect configuration of STCN on the edge leads to a loop in the network topology.
Step 7	**rep block port** {**id** *port-id* | *neighbor-offset* | **preferred**} **vlan** {*vlan-list* | **all**} **Example:** `Device(config-if)# rep block port id 0009001818D68700 vlan 1-100`	(Optional) Configures VLAN load balancing on the primary edge port, identifies the REP alternate port in one of three ways (**id** *port-id*, *neighbor_offset*, **preferred**), and configures the VLANs to be blocked on the alternate port. • **id** *port-id*: Identifies the alternate port by port ID. The port ID is automatically generated for each port in the segment. You can view interface port IDs by entering the **show interface** *type number* **rep** [**detail**] privileged EXEC command. • *neighbor_offset*: Number to identify the alternate port as a downstream neighbor from an edge port. The range is from -256 to 256, with negative numbers indicating the downstream neighbor from the secondary edge port. A value of **0** is invalid. Enter **-1** to identify the secondary edge port as the alternate port. **Note** Because you enter the **rep block port** command at the primary edge port (offset number 1), you cannot enter an offset value of 1 to identify an alternate port. • **preferred**: Selects the regular segment port previously identified as the preferred alternate port for VLAN load balancing. • **vlan** *vlan-list*: Blocks one VLAN or a range of VLANs. • **vlan all**: Blocks all the VLANs. **Note** Enter this command only on the REP primary edge port.
Step 8	**rep preempt delay** *seconds*	(Optional) Configures a pre-empt time delay.

	Command or Action	Purpose
	Example: `Device(config-if)# rep preempt delay 100`	• Use this command if you want VLAN load balancing to be automatically triggered after a link failure and recovery. • The time delay range is between 15 to 300 seconds. The default is manual pre-emption with no time delay. **Note** Enter this command only on the REP primary edge port.
Step 9	**rep lsl-age-timer** *value* **Example:** `Device(config-if)# rep lsl-age-timer 2000`	(Optional) Configures a time (in milliseconds) for which the REP interface remains up without receiving a hello from a neighbor. The range is from 120 to 10,000 ms in 40-ms increments. The default is 5000 ms (5 seconds). **Note** • EtherChannel port channel interfaces do not support LSL age-timer values that are less than 1000 ms. • Ensure that both the ports on the link have the same LSL age configured in order to avoid link flaps.
Step 10	**end** **Example:** `Device(config-if)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 11	**show interface** [*interface-id*] **rep** [**detail**] **Example:** `Device# show interface gigabitethernet1/1 rep detail`	(Optional) Displays the REP interface configuration.
Step 12	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the router startup configuration file.

Setting Manual Preemption for VLAN Load Balancing

If you do not enter the **rep preempt delay** *seconds* interface configuration command on the primary edge port to configure a preemption time delay, the default is to manually trigger VLAN load balancing on the segment. Be sure that all the other segment configurations have been completed before manually preempting

VLAN load balancing. When you enter the **rep preempt delay segment** *segment-id* command, a confirmation message is displayed before the command is executed because preemption might cause network disruption.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**rep preempt segment** *segment-id* **Example:** `Device# rep preempt segment 100` `The command will cause a momentary traffic disruption.` `Do you still want to continue? [confirm]`	Manually triggers VLAN load balancing on the segment. You need to confirm the command before it is executed.
Step 3	**show rep topology segment** *segment-id* **Example:** `Device# show rep topology segment 100`	(Optional) Displays REP topology information.
Step 4	**end** **Example:** `Device# end`	Exits privileged EXEC mode.

Configuring SNMP Traps for REP

You can configure a router to send REP-specific traps to notify the Simple Network Management Protocol (SNMP) server of link-operational status changes and port role changes.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib rep trap-rate** *value* **Example:** `Device(config)# snmp mib rep trap-rate 500`	Enables the switch to send REP traps, and sets the number of traps sent per second. • Enter the number of traps sent per second. The range is from 0 to 1000. The default

	Command or Action	Purpose
		is 0 (no limit is imposed; a trap is sent at every occurrence).
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**show running-config** **Example:** `Device# show running-config`	(Optional) Displays the running configuration, which can be used to verify the REP trap configuration.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the switch startup configuration file.

Configuring REP ZTP

To configure REP ZTP, you enable or disable it at the global level and the interface level. The default states are:

- Global level: Enabled
- Interface level: Disabled

You must explicitly enable the feature at the interface level on the upstream device interface connected to the downstream device. When enabled, only that interface will receive notification from the downstream switch to block or unblock the PnP startup VLAN.

Procedure

Step 1 Enter global configuration mode:

```
Switch# configure terminal
```

Step 2 Globally enable REP ZTP:

```
Switch(config)# rep ztp
```

Use the no form of the command to disable REP ZTP: `Switch(config)# no rep ztp`

Step 3 Enter interface configuration mode on the upstream device interface that is connected to the downstream device:

```
Switch(config)# interface <interface-name>
```

Step 4 Enable REP ZTP on the interface:

```
Switch(config-if)#rep ztp-enable
```

Use the no form of the command to disable REP ZTP on the interface: `Switch(config-if)#no rep ztp-enable`

Example

The following example shows the minimum configuration required to enable the REP ZTP feature on the upstream device interface that is connected to a downstream device.

```
Switch#show running-config interface gigabitEthernet 1/1
Building configuration...

Current configuration : 93 bytes
!
interface GigabitEthernet1/1
 switchport mode trunk
 rep segment 100
 rep ztp-enable
end
```

Monitoring Resilient Ethernet Protocol Configurations

This is an example of the output for the **show interface** [*interface-id*] **rep** [**detail**] command. This display shows the REP configuration and status on an uplink port.

```
Device# show interfaces GigabitEthernet1/1 rep detail

GigabitEthernet1/1 REP enabled
Segment-id: 3 (Primary Edge)
PortID: 03010015FA66FF80
Preferred flag: No
Operational Link Status: TWO_WAY
Current Key: 02040015FA66FF804050
Port Role: Open
Blocked VLAN: <empty>
Admin-vlan: 1
REP-ZTP Status: Disabled
Preempt Delay Timer: disabled
Configured Load-balancing Block Port: none
Configured Load-balancing Block VLAN: none
STCN Propagate to: none
LSL PDU rx: 999, tx: 652
HFL PDU rx: 0, tx: 0
BPA TLV rx: 500, tx: 4
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 6, tx: 5
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 135, tx: 136
```

This is an example of the output for the **show interface** [*interface-id*] **rep** [**detail**] command. This display shows the REP configuration and status on a downlink port.

```
Device#show interface GigabitEthernet1/1 rep detail
GigabitEthernet1/1   REP enabled
Segment-id: 1 (Segment)
```

```
PortID: 019B380E4D9ACAC0
Preferred flag: No
Operational Link Status: NO_NEIGHBOR
Current Key: 019B380E4D9ACAC0696B
Port Role: Fail No Ext Neighbor
Blocked VLAN: 1-4094
Admin-vlan: 1
REP-ZTP Status: Disabled
Preempt Delay Timer: 100 sec
LSL Ageout Timer: 2000 ms
LSL Ageout Retries: 5
Configured Load-balancing Block Port: 09E9380E4D9ACAC0
Configured Load-balancing Block VLAN: 1-100
STCN Propagate to: segment 25
LSL PDU rx: 292, tx: 340
HFL PDU rx: 0, tx: 0
BPA TLV rx: 0, tx: 0
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 0, tx: 0
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 0, tx: 0
```

This is an example for the **show rep topology** [**segment** *segment-id*] [**archive**] [**detail**] command. This display shows the REP topology information for all the segments.

```
Device# show rep topology

REP Segment 1
BridgeName       PortName     Edge Role
---------------- ----------   ---- ----
10.64.106.63     Gi1/4        Pri  Open
10.64.106.228    Gi1/4             Open
10.64.106.228    Gi1/3             Open
10.64.106.67     Gi1/3             Open
10.64.106.67     Gi1/4             Alt
10.64.106.63     Gi1/4        Sec  Open

REP Segment 3
BridgeName       PortName     Edge Role
---------------- ----------   ---- ----
10.64.106.63     Gi1/1        Pri  Open
SVT_3400_2       Gi1/3             Open
SVT_3400_2       Gi1/4             Open
10.64.106.68     Gi1/2             Open
10.64.106.68     Gi1/1             Open
10.64.106.63     Gi1/2        Sec  Alt
```

Displaying REP ZTP Status

Use the **show** command to identify the state of REP ZTP on an interface. In the following example, the feature is disabled on interface GigabitEthernet 1/1 and it is enabled on interface GigabitEthernet 1/1. The status of **pnp_startup_vlan** is "Blocked".

Procedure

Step 1 In priviledged exec mode, enter:

show interfaces rep detail

Example:

```
GigabitEthernet1/1   REP enabled
Segment-id: 100 (Segment)
PortID: 00016C13D5AC4320
Preferred flag: No
Operational Link Status: TWO_WAY
Current Key: 00026C13D5AC43209DAB
Port Role: Open
Blocked VLAN: <empty>
Admin-vlan: 1
REP-ZTP Status: Disabled
REP Segment Id Auto Discovery Status: Enabled
REP Segment Id Type: Manual
Preempt Delay Timer: disabled
LSL Ageout Timer: 5000 ms
LSL Ageout Retries: 5
Configured Load-balancing Block Port: none
Configured Load-balancing Block VLAN: none
STCN Propagate to: none
LSL PDU rx: 382, tx: 297
HFL PDU rx: 0, tx: 0
BPA TLV rx: 1, tx: 19
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 95, tx: 0
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 95, tx: 95

GigabitEthernet1/1   REP enabled
Segment-id: 100 (Segment)
PortID: 00026C13D5AC4320
Preferred flag: No
Operational Link Status: NO_NEIGHBOR
Current Key: 00026C13D5AC43209DAB
Port Role: Fail No Ext Neighbor
Blocked VLAN: 1-4094
Admin-vlan: 1
REP-ZTP Status: Enabled
REP-ZTP PnP Status: Unknown
REP-ZTP PnP Vlan: 1
REP-ZTP Port Status: Blocked
REP Segment Id Auto Discovery Status: Enabled
REP Segment Id Type: Manual
Preempt Delay Timer: disabled
LSL Ageout Timer: 5000 ms
LSL Ageout Retries: 5
Configured Load-balancing Block Port: none
Configured Load-balancing Block VLAN: none
STCN Propagate to: none
LSL PDU rx: 11, tx: 11
HFL PDU rx: 0, tx: 0
BPA TLV rx: 0, tx: 0
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
```

```
EPA-ELECTION TLV rx: 0, tx: 0
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 0, tx: 0
```

Step 2 Use the show command again to display the status of **pnp_startup_vlan**.

When the downstream device is booted up, it sends notification to the connected upstream switch interface to unblock the **pnp_startup_vlan** for it to get the DHCP IP address and further establish communication with the PNP server or Cisco Catalyst Center. The show command indicates the status as "Unblocked".

The following syslogs on the upstream switch notify you about FWD and BLK of ports. There are no syslogs in the downstream switch as PnP takes control of the console and no syslogs can be printed on the console.

```
REP-6-ZTPPORTFWD: Interface GigabitEthernet1/1 moved to forwarding on ZTP
notification
```

```
REP-6-ZTPPORTBLK: Interface GigabitEthernet1/1 moved to blocking on ZTP
notification
```

Example:

```
Switch#show interfaces rep detail
GigabitEthernet1/1   REP enabled
Segment-id: 100 (Segment)
PortID: 00016C13D5AC4320
Preferred flag: No
Operational Link Status: TWO_WAY
Current Key: 00026C13D5AC43209DAB
Port Role: Open
Blocked VLAN: <empty>
Admin-vlan: 1
REP-ZTP Status: Disabled
REP Segment Id Auto Discovery Status: Enabled
REP Segment Id Type: Manual
Preempt Delay Timer: disabled
LSL Ageout Timer: 5000 ms
LSL Ageout Retries: 5
Configured Load-balancing Block Port: none
Configured Load-balancing Block VLAN: none
STCN Propagate to: none
LSL PDU rx: 430, tx: 358
HFL PDU rx: 0, tx: 0
BPA TLV rx: 1, tx: 67
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 107, tx: 0
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 107, tx: 108

GigabitEthernet1/1   REP enabled
Segment-id: 100 (Segment)
PortID: 00026C13D5AC4320
Preferred flag: No
Operational Link Status: NO_NEIGHBOR
Current Key: 00026C13D5AC43209DAB
Port Role: Fail No Ext Neighbor
Blocked VLAN: 1-4094
Admin-vlan: 1
REP-ZTP Status: Enabled
REP-ZTP PnP Status: In-Progress
REP-ZTP PnP Vlan: 69
REP-ZTP Port Status: Unblocked
```

```
REP Segment Id Auto Discovery Status: Enabled
REP Segment Id Type: Manual
Preempt Delay Timer: disabled
LSL Ageout Timer: 5000 ms
LSL Ageout Retries: 5
Configured Load-balancing Block Port: none
Configured Load-balancing Block VLAN: none
STCN Propagate to: none
LSL PDU rx: 32, tx: 40
HFL PDU rx: 0, tx: 0
BPA TLV rx: 0, tx: 0
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 0, tx: 0
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 0, tx: 0
```

Step 3 Use the **show platform hardware fed switch active vlan** *vlan-id* command to check the interface state of the PnP startup VLAN:

Example:

```
Switch#show platform hardware fed switch active vlan 901
vlan id is:: 901
Interfaces in forwarding state: : Gig1/1(Untagged), Gig1/2(Untagged)
flood list: : Gig1/1, Gig1/2
```

Step 4 (Optional) Use the following debug commands to troubleshoot REP ZTP:

- **debug rep lslsm**: This command helps you understand LSL state machine events in the NO_NEIGHBOR state.
- **debug rep packet**: Use this command to dump LSL packets with the REP ZTP LSL TLV to check the PnP status on the peer client node.

CHAPTER 174

Media Redundancy Protocol

Media Redundancy Protocol

Media Redundancy Protocol (MRP), defined in International Electrotechnical Commission (IEC) standard 62439-2, provides fast convergence in a ring network topology for Industrial Automation networks. MRP Media Redundancy Manager (MRM) defines its maximum recovery times for a ring in the following range: 200 ms and 500 ms.

Note The default maximum recovery time on the Cisco IE switch is 200 ms for a ring composed of up to 50 nodes. You can configure the switch to use the 500 ms recovery time profile as described in Configure MRP Auto-Manager, on page 2607. The 10 ms and 30 ms recovery time profiles are not supported.

MRP operates at the MAC layer and is commonly used in conjunction with the PROFINET standard for industrial networking in manufacturing.

MRP Mode

MRP is supported on the switches.

MRP CLI mode is managed by the Cisco IOS XE CLI and WebUI, a web-based user interface (UI).

Note When managing the switch in MRP CLI mode, you cannot download the MRP configuration from Siemens STEP7/TIA.

Protocol Operation

In an MRP ring, the MRM serves as the ring manager, while the Media Redundancy Clients (MRCs) act as member nodes of the ring. Each node (MRM or MRC) has a pair of ports to participate in the ring. The MRM initiates and controls the ring topology to react to network faults by sending control frames on one ring port over the ring and receiving them from the ring over its other ring port, and conversely in the other direction. An MRC reacts to received reconfiguration frames from the MRM and can detect and signal link changes on its ring ports.

On the switch, certain nodes or all nodes in the ring can also be configured to start as a Media Redundancy Automanager (MRA). MRAs select one MRM among each other by using a voting protocol and a configured priority value. The remaining MRAs transition to the MRC role.

All MRM and MRC ring ports support the following states:

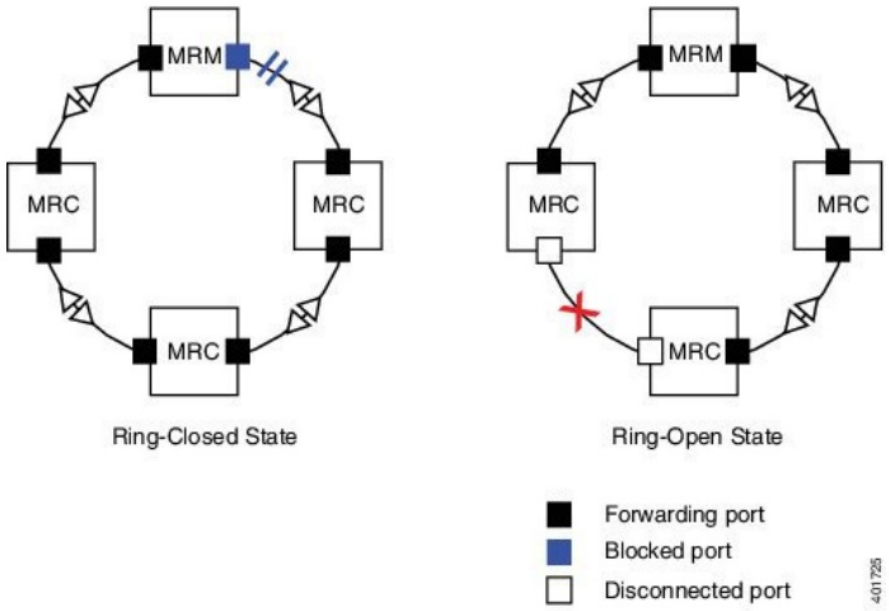
- Disabled: Ring ports drop all received frames.
- Blocked: Ring ports drop all received frames except MRP control frames and some standard frames, for example, LLDP.
- Forwarding: Ring ports forward all received frames.
- Not Connected: The link is physically down or disconnected. (This state differs from the Disabled state, in which the MRP Port is manually disabled through software.)

During normal operation, the network operates in the Ring-Closed state (see figure below). To prevent a loop, one of the MRM ring ports is blocked, while the other port is forwarding. Most of the time, both ring ports of all MRCs are in the forwarding state. With this loop avoidance, the physical ring topology becomes a logical stub topology.

In the figure, note the following details about the two rings, left and right:

- Left Ring: The connection (small blue square, top) on the MRM is in a blocked state (as shown by the two parallel lines) because no ports are disconnected.
- Right Ring: Two MRC connections (left and center small white squares) are in the disabled state because the link between them is broken, as marked by a red “x”.

Figure 176: MRP Ring States



If a network failure occurs:

- The network shifts into the Ring-Open state.
- In the case of failure of a link connecting two MRCs, both ring ports of the MRM change to the forwarding state, the MRCs adjacent to the failure have a disabled and a forwarding ring port, and the other MRCs have both ring ports forwarding.

In the Ring-Open state, the network logical topology becomes a stub.

Layer 2 Ethernet frames will be lost during the time required for the transition between these two ring states. The MRP protocol defines the procedures to automatically manage the switchover to minimize the switchover time. A recovery time profile, composed of various parameters, drives the MRP topology convergence performance. The 200 ms profile supports a maximum recovery time of 200 ms.

MRP uses three types of control frames:

- To monitor the ring status, MRM regularly sends test frames on both ring ports.
- When MRM detects failure or recovery, it sends TopoChange frames on both ring ports.
- When MRC detects failure or recovery on a local port, it sends LinkChange subtype frames, Linkdown and Linkup, to the MRM.

Media Redundancy Automanager

If configured to start as a Media Redundancy Automanager (MRA), the node or nodes select an MRM using a voting protocol and configured priority value. The remaining MRAs transition to the MRC role. All nodes must be configured as MRA. A manually configured MRM and MRA in the same ring is not supported.

Note

- You can activate MRA through the CLI. See the section Configuring MRP CLI Mode, on page 2607 in this guide.
- Although MRAs transition to the MRC role after an MRM is selected, you cannot explicitly configure an MRC.

The MRA role is not an operational MRP role like MRM or MRC. It is only an administrative, temporary role at device startup, and a node must transition to the MRM role or the MRC role after startup and the MRM is selected though the manager voting process.

MRA functions as follows:

1. At power on, all MRAs begin the manager voting process. Each MRA begins to send MRP_Test frames on both ring ports. The MRP_Test frame contains the MRA's priority value. The remote manager's priority value contained in the received MRP_Test frames are compared with the MRA's own priority. If its own priority is higher than the received priority, the MRA sends a negative test manager acknowledgment (MRP_TestMgrNAck) frame, along with the remote manager's MAC address.
2. If the receiving MRA receives an MRP_TestMgrNAck with its own MAC address, the receiving MRA initiates the transition into the client (MRC) role.
3. The MRP_TestPropagate frame informs other MRA devices in the client role about the role change and the new higher priority manager. The clients receiving this frame update their higher priority manager information accordingly. This ensures that clients remain in the client role if the monitored higher priority manager role changes.

Licensing

You do not need a feature license to use MRP with Switches. MRP works with either base license—Network Essentials or Network Advantage.

To find information about platform support and to know which license levels a feature is available with, use Cisco Feature Navigator. To access Cisco Feature Navigator, go to https://www.cisco.com/go/cfn. An account on cisco.com is not required.

Multiple MRP Rings

In an Industrial Ethernet network, an MRP ring in a cell/area is a sub-ring of the access layer. You can connect multiple MRP rings, which you can then aggregate into the distribution layer.

You can configure up to 3 rings. The MRP switch can be configured only as an auto-manager.

MRP-STP Interoperability

MRP works with Spanning Tree Protocol (STP) to prevent unwanted broadcast loops in the event that a user accidentally connects a device that does not participate in the MRP ring. In a network operating with MRP

and STP, spanning tree bridge protocol data units (BPDUs) are not sent on MRP-enabled ports. If ports are unconfigured from an MRP ring, then the ports are added to the spanning tree.

MRP-STP interoperability is supported in MRP CLI mode, and functions without additional CLI configuration.

Prerequisites

- Because MRP is deployed in a physical Ring topology, before configuring or unconfiguring the MRP feature, it is advised to leave one physical connection between two nodes in each ring open by either issuing a **shut** command on the connecting interfaces or physically removing the cable to avoid any network storms. After you have properly configured all MRMs, issue a **no shut** command on the port or re-connect the cable between the nodes.

Guidelines and Limitations

General Guidelines and Limitations

- To avoid Smart License registration failure, ensure that the NTP configuration and the device clock are in sync.
- Support for multiple MRP rings is available only through the CLI or WebUI.
- The switch supports up to 50 MRCs per ring.
- MRP cannot run on the same interface (port) as Resilient Ethernet Protocol (REP), Device Level Ring (DLR), Spanning Tree Protocol (STP), Flex Links, macsec, or Dot1x.
- For access ports, you must specifically configure **switchport mode access** and **switchport access vlan** *x* commands in the MRP interface.
- MRP interfaces come up in a forwarding state and remain in a forwarding state until notified that it is safe to block. The MRP ring state changes to Ring-Closed.
- MRP ports cannot be configured as any of these port types: SPAN destination port, Private VLAN port, or Tunnel port.
- MRP is not supported on EtherChannels or on an individual port that belongs to an EtherChannel.
- Each MRP ring can have one MRP VLAN. The VLAN must be different for each ring in a device to avoid traffic flooding.

MRP CLI Mode Guidelines and Limitations

- After using the CLI to configure the MRP ring, you must attach the MRP ring to a pair of ports that support MRP.
- Both MRP ports must have the same interface mode (access or trunk).
- To change an existing MRP ring's configuration (mode), or to change the interface mode of the ring ports between access and trunk, you must first delete the ring and then recreate it with the new configuration.

- When both MRP ports are in access mode, the access VLANs should match. If the configured MRP VLAN does not match the ports' access VLAN, the MRP VLAN is automatically changed to the MRP ports' access VLAN.
- In an MRP ring with two access ports, if the ports do not belong to the same access VLAN when you create the MRP ring or you change the access VLAN for only one of the ports after the MRP ring is created, the MRP ring operation is suspended and a message similar to the following is displayed:

```
ERROR% The ring 1 ports don't belong to the same access VLAN. The MRP ring will not
function until the issue has been fixed
```

  Resolve the issue by configuring the access VLAN to be the same for the two ring ports.
- The 200 ms standard profile and 500 ms profile are supported. The 10 ms profile and 30 ms profile are not supported.
- You can activate MRA through the CLI.
- Although MRAs transition to the MRC role after an MRM is selected, you cannot explicitly configure an MRC.

Default Settings

- MRP is disabled by default; MRP CLI is the default mode when MRP is enabled.
- The default VLAN is 1.

Note Create the non-default VLAN before you assign it to MRP ring 1.

Guidelines and limitations to PROFINET MRP mode configuration

Before configuring the Cisco switch with PROFINET MRP through Siemens TIA or STEP7, ensure the following:

- PROFINET MRP feature doesn't support MRC role.
- Use the TIA portal to configure or modify MRA role.
- Avoid using CLI for configuration when TIA is in use. MRP CLI mode and PROFINET MRP mode are mutually exclusive.
- If the IE3500 switch is connected to the PROFINET PLC, the output of **show profinet status | include Connected** should display **Yes**. If it displays **No**, the switch is not connected to the PROFINET PLC.
- Ensure that the GSD file version matches the Cisco IOS release to avoid compatibility issues. For detailed configuration steps, refer to the PROFINET Protocol Configuration Guide.

Install the PROFINET GSD File

The PROFINET MRP GSD file is bundled with the Cisco IOS XE software release. After the switch boots at least one time, the GSD files for the switch are located in a directory called "ProfinetGSD". In this directory, there is a zip file containing all the GSDs for all the switch SKUs. Use the GSD file bundled with the release and included in the ProfinetGSD directory.

Note Remove the older GSD XML file from TIA 15 or STEP 7 to ensure compatibility with the Cisco IOS software.

Configure PROFINET MRP

This task guides you through configuring PROFINET MRP to ensure proper network operation and redundancy.

Before you begin

Disconnect an MRP Ethernet port from the ring (open ring) to discover all neighboring devices using the LLDP protocol. Perform this step before deploying PROFINET MRP to the network. This approach prevents unnecessary flooding if configuration issues occur.

- (Optional) Verify **Neighbor Discovery** with LLDP.

  Use the command **show lldp neighbor** to confirm all neighbor devices are correctly discovered before continuing with PROFINET MRP setup.

- Check that the PROFINET status indicates a **connected-state**.

- Inspect the MRP ring port status:

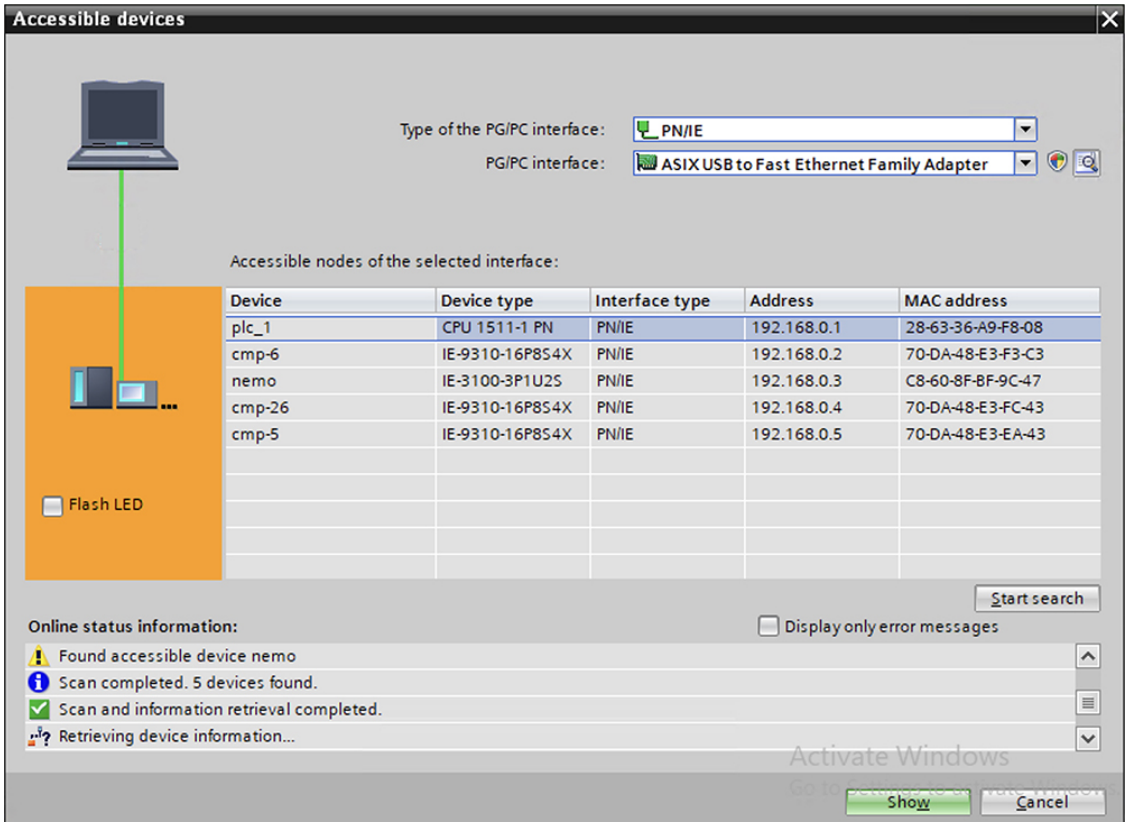
  Use the **profinet mrp ring 1** command.

Procedure

Step 1 Access the PROFINET Device Discovery (DCP) window.

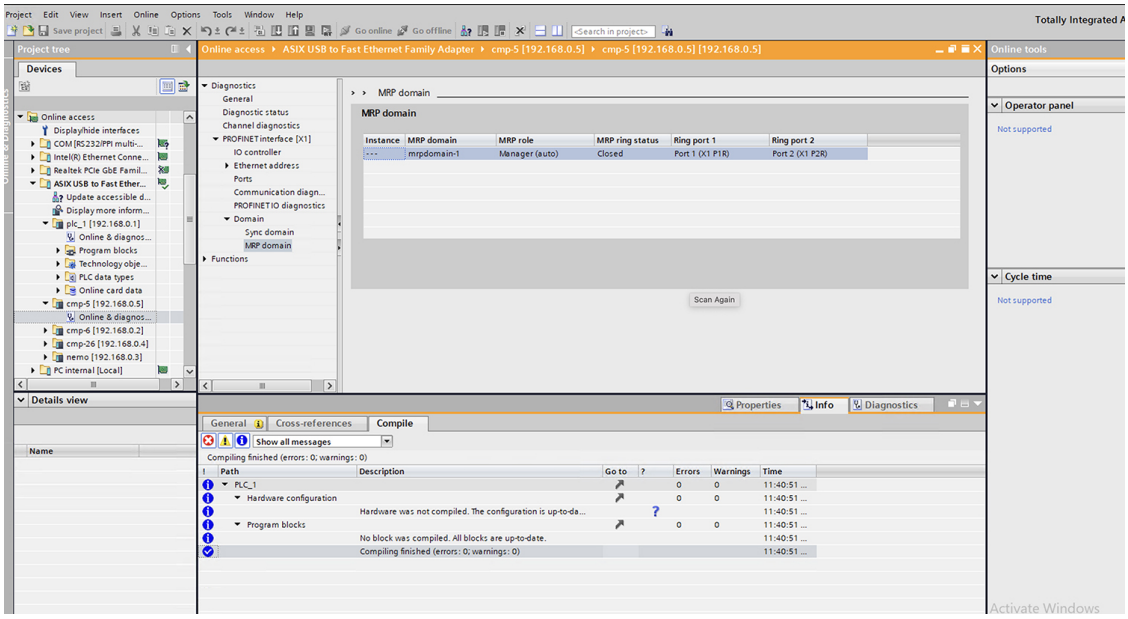
Open the PROFINET DCP window to identify and manage devices in the network.

Figure 177: PROFINET Device Discovery (DCP) window before configuring MRP



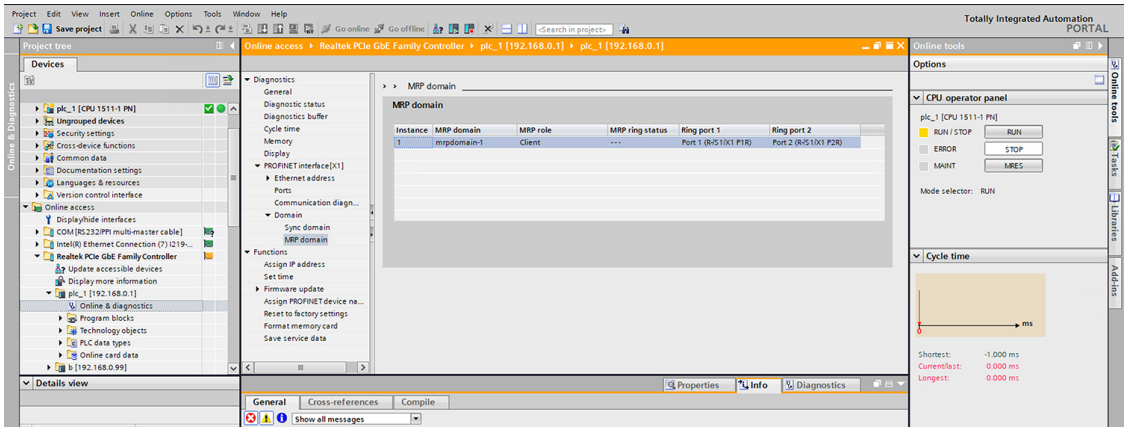
Step 2 Assign PROFINET MRP Manager role and domain name on MRM device.

Figure 178: PROFINET MRP Manager role and MRP domain name



Step 3 Define the PROFINET MRP client and MRP domain name on client devices.

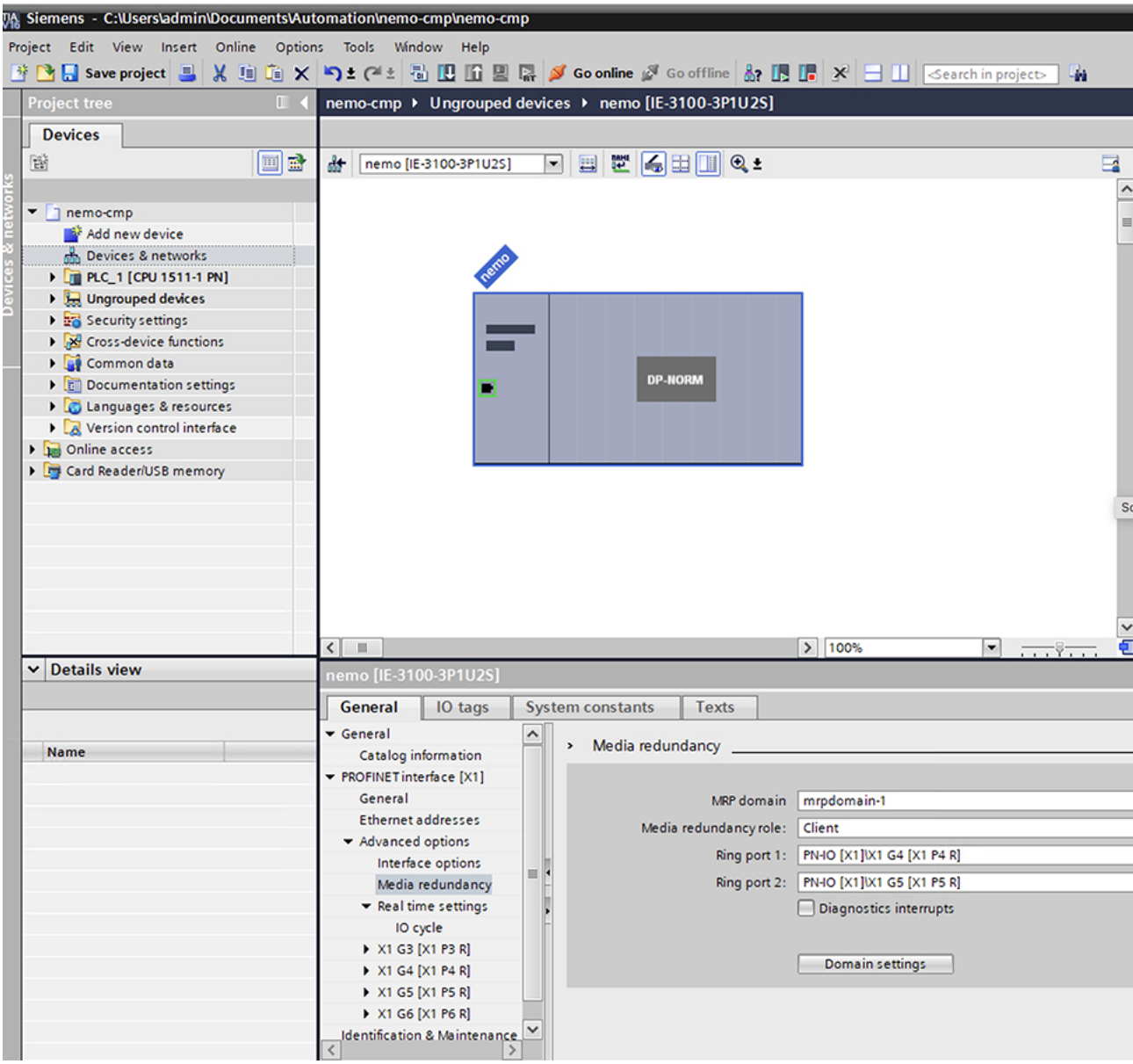
Figure 179: PROFINET MRP and MRP domain on client



Step 4 When using MRA mode, configure all devices and domain details.

Step 5 Configure the PROFINET MRP interfaces on all devices participating in the ring.

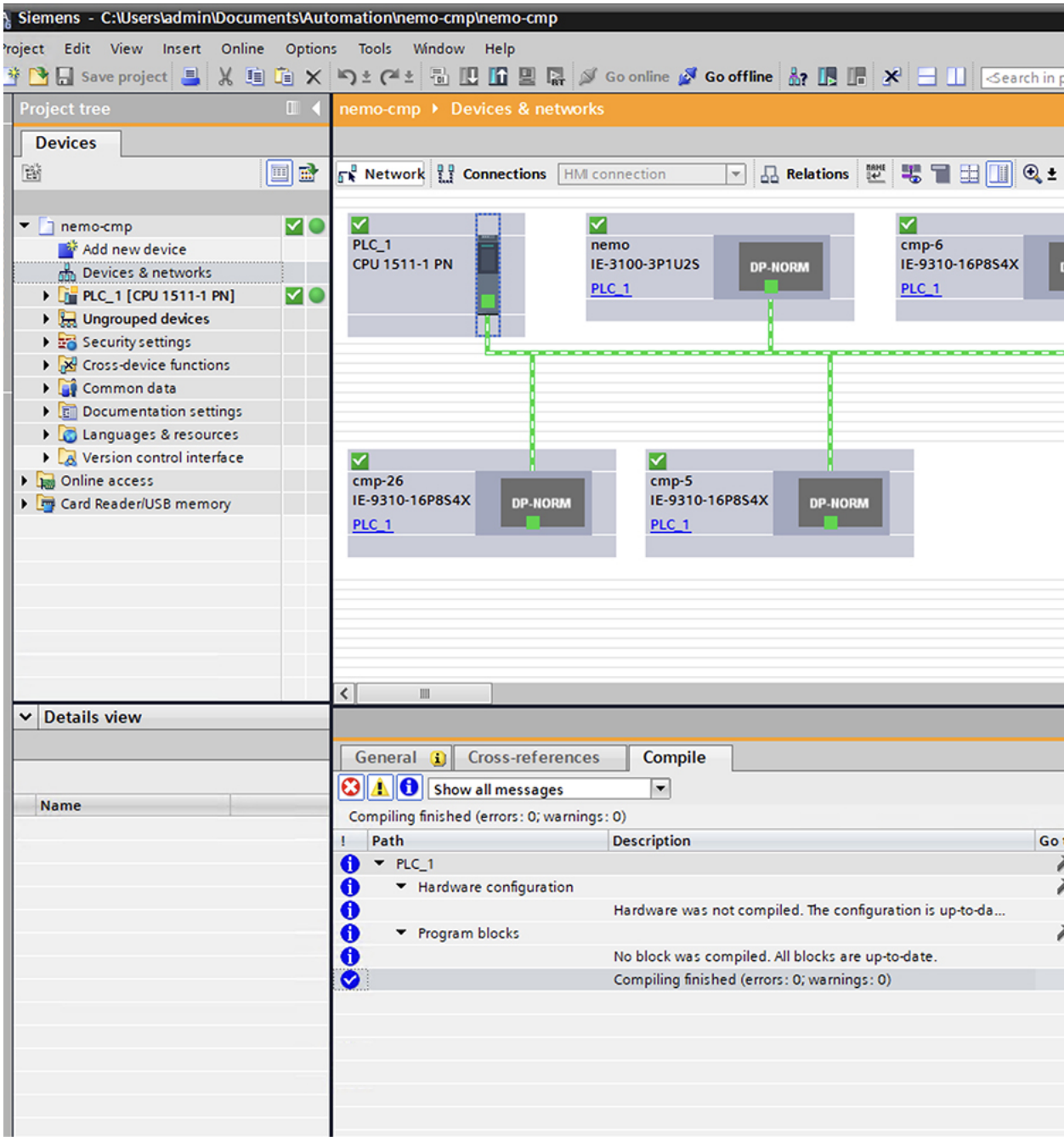
Figure 180: PROFINET MRP inerfaces



Step 6 Compile the configuration, and then download it to the PLC device.

Step 7 Verify that all devices are connected and the MRP ring is closed.

Figure 181: PROFINET MRP network configuration diagram



Step 8 Verify on the devices that the MRP ring is closed by using the **show profinet mrp ring 1** command:

Ensure that one port is in the **Blocked** state and the other port is in the **Forwarding** state.

Managing PROFINET Using Simatic Step 7 or TIA 15 Portal

This section provides an overview of key screens within the TIA portal. It does not provide any configuration details. For details on using the TIA portal, refer to the Siemens Simatic STEP7 user documentation.

Note MRP automanager in PROFINET mode is supported only in TIA V15.

Figure 182: PROFINET Device Discovery (DCP) Window Before Configuring MRP

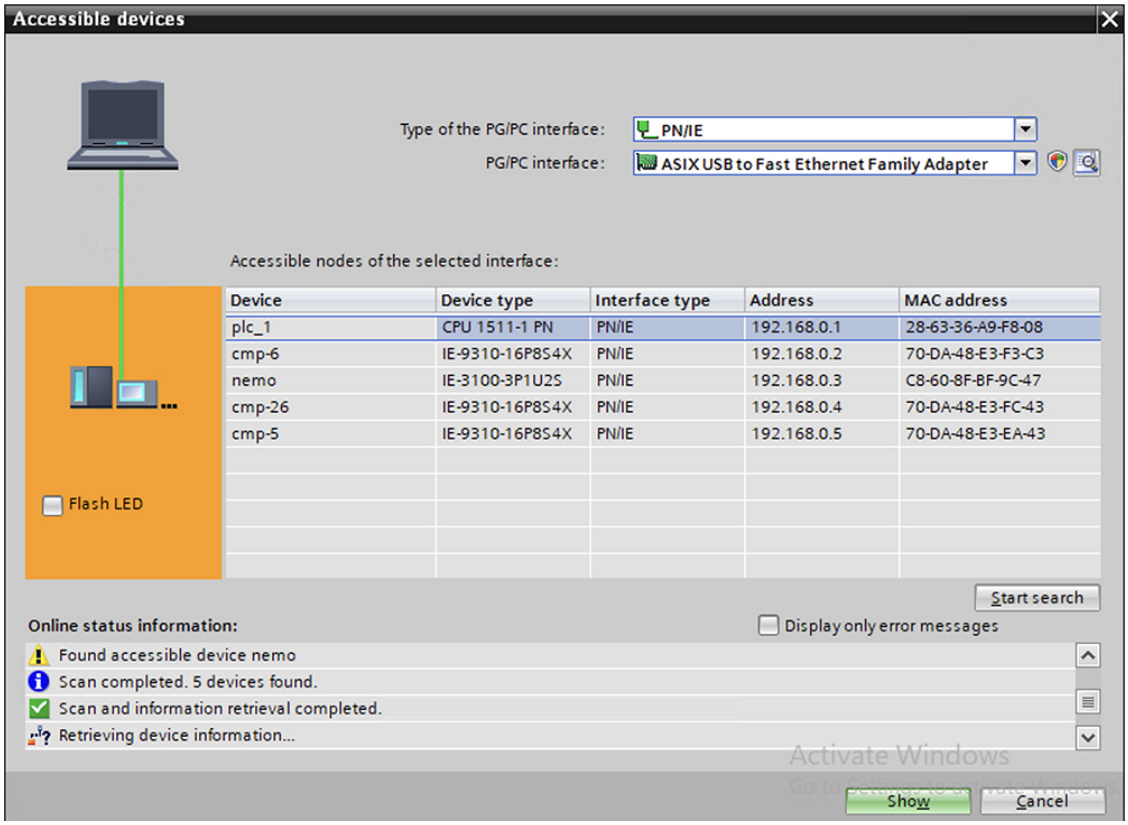


Figure 183: Define PROFINET MRP Manager and MRP domain

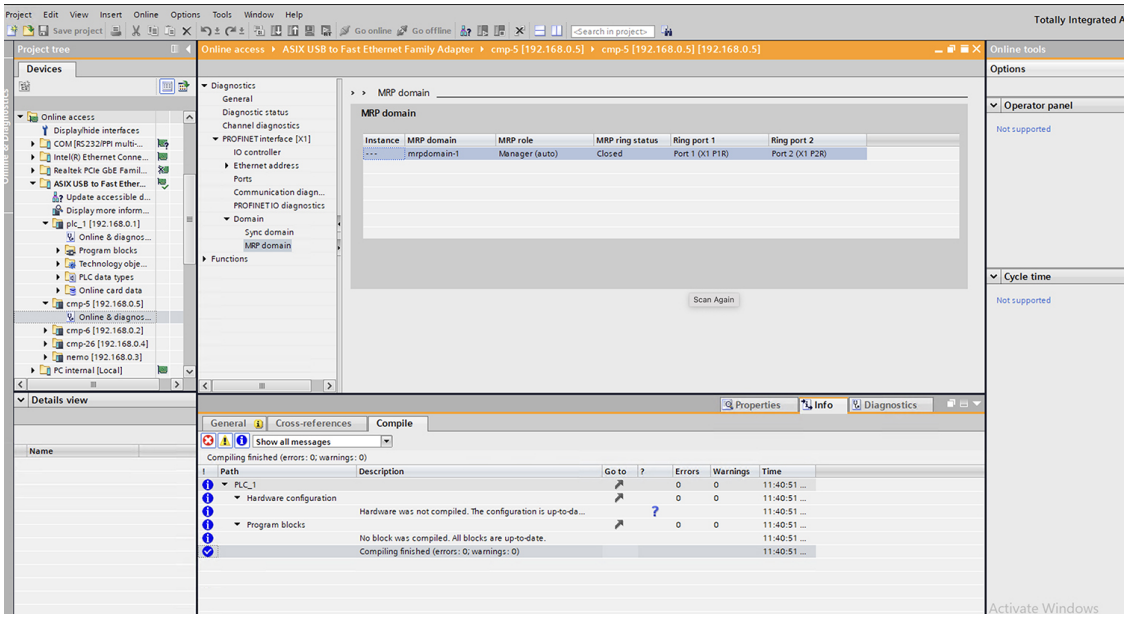


Figure 184: Define PROFINET MRP Client and MRP Domain

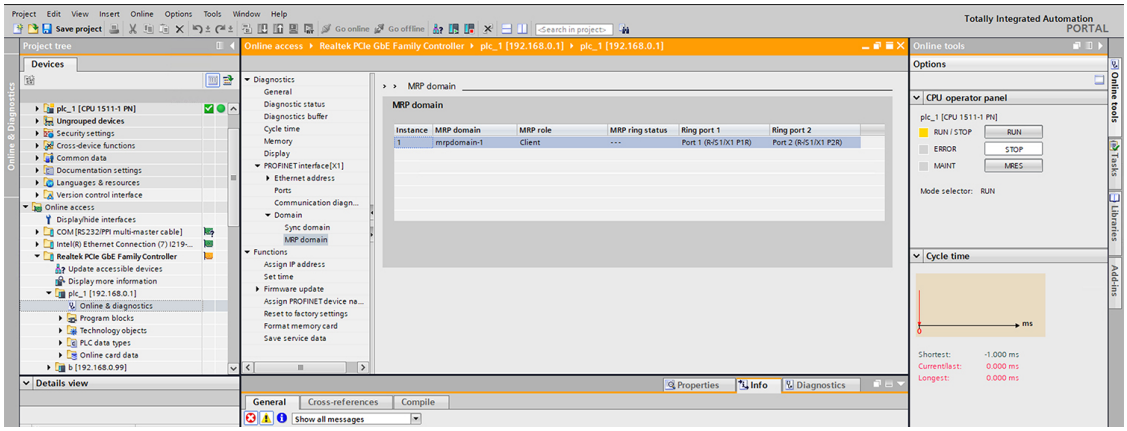


Figure 185: Define PROFINET MRP Interfaces

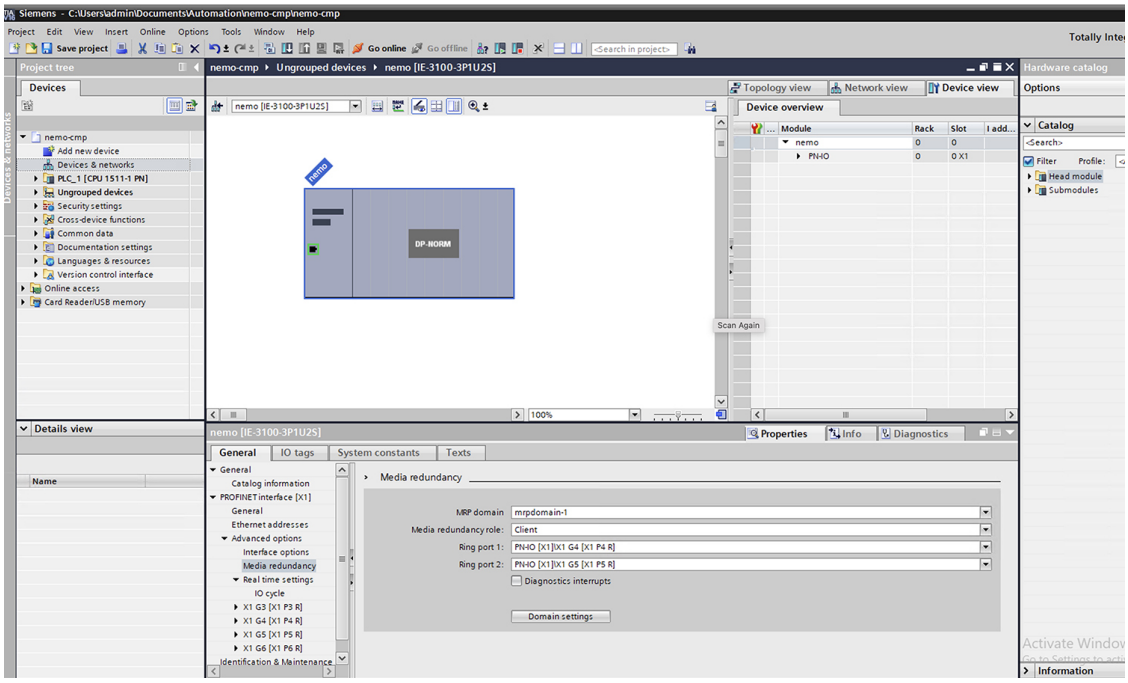
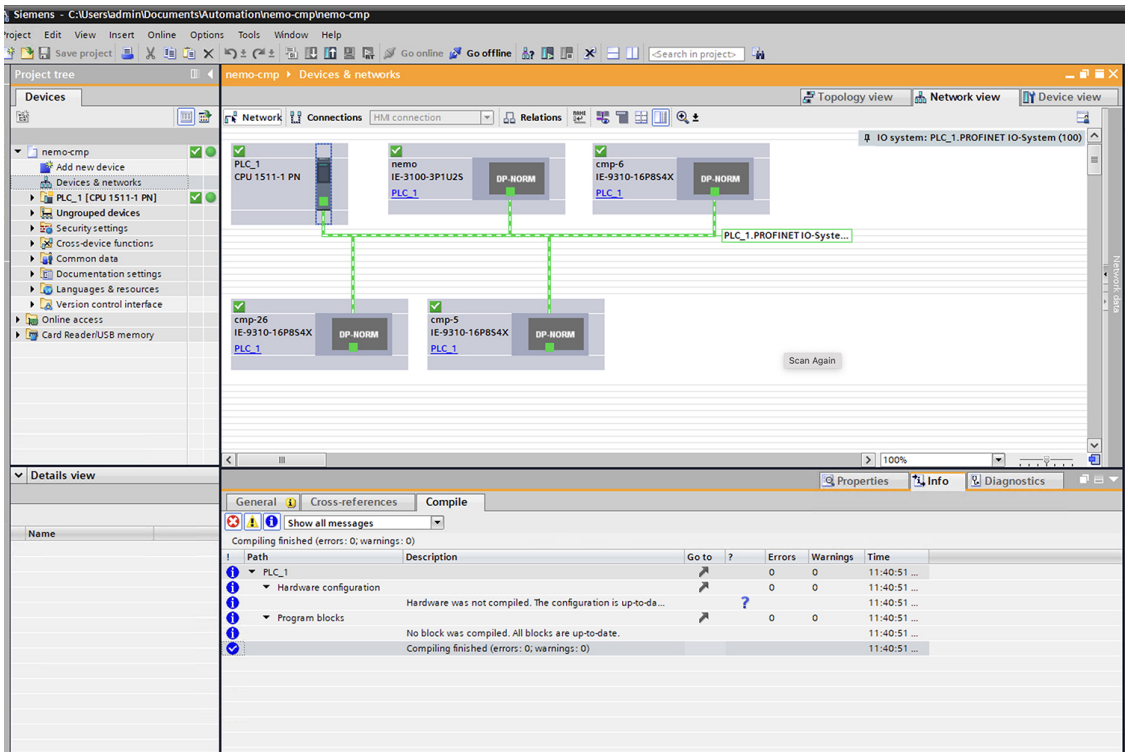


Figure 186: PROFINET MRP Network Configuration Diagram



Configuring MRP CLI Mode

To configure MRP, configure the node as MRA and specify the two MRP ports. You can configure up to 3 rings on the device (the device can be manager or client) with a manager instance for each ring and one manager per device.

The following MRP configuration parameters are optional:

- domain-id: A unique ID that represents the MRP ring.
- domain-name: Logical name of the configured MRP domain-ID.
- profile: 200 ms (the default)
- vlan-id: VLAN for sending MRP frames.

Configure MRP Auto-Manager

Follow this procedure to configure the switch as MRA in MRP CLI mode, which is the default.

Note If the device is connected to a PLC module, please make sure “no device in the ring” is selected for MRP.

Procedure

Step 1 Enable MRP:

mrp ring *mrp_id*

MRP supports up to 3 rings.

Step 2 Configure MRP auto-manager mode on the switch:

mode auto-manager

Step 3 (Optional for single MRP ring) Configure the domain ID:

domain-id *value*

value: UUID string of 32 hexadecimal digits in five groups separated by hyphens

Example: 550e8400-e29b-41d4-a716-446655440000

The default domain ID for ring 1 is FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFE.

Note
Only change the domain-ID from the default when required.

Step 4 (Optional for single MRP ring) Configure the domain name:

domain-name *name*

name: String of up to 32 characters

Step 5 (Optional) Configure the VLAN ID:

vlan-id *vlan*

Step 6 (Optional) Configure the recovery profile:

profile { |**200** | **500**}

- 200: Maximum recovery time 200 milliseconds
- 500; Maximum recovery time 500 milliseconds

Step 7 (Optional) Configure the MRA priority:

priority *value*

value: Range <36864 – 61440>, lowest: 65535.

The default priority is 40960.

Step 8 (Optional) Configure the interval:

interval *interval*

Note
The Interval field is not displayed in WebUI for MRP.

- 3: 3 milliseconds MRP_Test default interval for 30 ms profile
- 20: 20 milliseconds MRP_Test default interval for 200 ms profile
- 50: 50 milliseconds MRP_Test default interval for 500 ms profile
- <3-10>: Optional faster MRP_Test interval in milliseconds

Note
The optional faster MRP_Test interval can be configured only when the ring is formed with IE3x00 devices.

Step 9 Specify the ID of the port that serves as the first ring port:

interface *port*

Step 10 Configure the interface mode:

switchport mode { **access** | **trunk** }

Note
You must specify **switchport mode access** when configuring MRP in access mode.

Step 11 Associate the interface to the MRP ring:

mrp ring 1

Step 12 Return to global configuration mode:

exit

Step 13 Specify the ID of the port that serves as second ring port:

interface *port*

Step 14 Configure the interface mode:

switchport mode { **access** | **trunk** }

Note
You must specify **switchport mode access** at this step when configuring MRP in access mode.

Step 15 Associate the interface to the MRP ring:

mrp ring 1

Step 16 Return to privileged EXEC mode:

end

Step 17 (For multiple rings) Repeat step 1 through 14 for each additional ring:

- Assign ring number 2 for the second ring.
- Assign a unique domain ID for Ring 2. The default domain ID for ring 2 is FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFD.
- Assign ring number 3 for the third ring.
- Assign a unique domain ID for Ring 3. The default domain ID for ring 3 is FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFC.

Note
Each ring should have its own domain ID. No two rings share the same domain ID.

Example

The following example shows configuring MRP automanager:

```
Switch#configure terminal
Switch# no profinet mrp
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)#mrp ring 1
Switch(config-mrp)#mode auto-manager
Switch(config-mrp-auto-manager)#domain-id FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF
Switch(config-mrp-auto-manager)#priority 40960
Switch(config-mrp-auto-manager)#end
Switch#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)#GigabitEthernet1/1
Switch(config-if)#switchport mode trunk
Switch(config-if)#mrp ring 1
WARNING% Enabling MRP automatically set STP FORWARDING. It is recommended to shutdown all
interfaces which are not currently in use to prevent potential bridging loops.
Switch(config-if)#exit
Switch(config)#GigabitEthernet1/1
Switch(config-if)#switchport mode trunk
Switch(config-if)#mrp ring 1
WARNING% Enabling MRP automatically set STP FORWARDING. It is recommended to shutdown all
interfaces which are not currently in use to prevent potential bridging loops.
Switch(config-if)#exit
Switch(config-if)#end
```

```
Switch# show mrp ring 1
MRP ring 1

Profile : 200 ms
Mode : Auto-Manager
Priority : 40960
Operational Mode: Client
From : CLI
License : Active
Best Manager :
MAC Address : 00:78:88:5E:03:81
Priority : 36864

Network Topology: Ring
Network Status : OPEN
Port1: Port2:
MAC Address :84:B8:02:ED:E8:02 MAC Address :84:B8:02:ED:E8:01
Interface :GigabitEthernet1/1 Interface :GigabitEthernet1/1
Status :Forwarding Status :Forwarding

VLAN ID : 1
Domain Name : Cisco MRP Ring 1
Domain ID : FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF

Topology Change Request Interval : 10ms
Topology Change Repeat Count : 3
Short Test Frame Interval : 10ms
Default Test Frame Interval : 20ms
Test Monitoring Interval Count : 3
Test Monitoring Extended Interval Count : N/A
Switch#show mrp ports

Ring ID : 1
PortName                Status
--------------------------------------
GigabitEthernet1/1         Forwarding
GigabitEthernet1/1         Forwarding
```

Note The **show mrp ring** output shows "License: Not Applicable" in CLI and Profinet mode.

Configuration Example

The following example shows the MRP switch configured as automanager:

```
Switch#configure terminal
Switch# no profinet mrp
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)#mrp ring 1
Switch(config-mrp)#mode auto-manager
Switch(config-mrp-auto-manager)#priority 36864
Switch(config-mrp-auto-manager)#end
Switch#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)#interface gi1/1
Switch(config-if)#switchport mode trunk
```

```
Switch(config-if)#mrp ring 1
WARNING% Enabling MRP automatically set STP FORWARDING. It is recommended to shutdown all
interfaces which are not currently in use to prevent potential bridging loops.
Switch(config-if)#exit
Switch(config)#interface gi1/1
Switch(config-if)#switchport mode trunk
Switch(config-if)#mrp ring 1
WARNING% Enabling MRP automatically set STP FORWARDING. It is recommended to shutdown all
interfaces which are not currently in use to prevent potential bridging loops.
Switch(config-if)#end

Switch#show mrp ring
MRP ring 1

Profile     : 200 ms
Mode        : Auto-Manager
Priority    : 36864
Operational Mode: Manager
From        : CLI
License     : Active
Best Manager MAC Address    :84:B8:02:ED:E8:01     priority 36864

Network Topology: Ring
Network Status  : OPEN
Port1:                                    Port2:
 MAC Address    :84:B8:02:ED:E8:02          MAC Address    :84:B8:02:ED:E8:01
 Interface      :GigabitEthernet1/1      Interface      :GigabitEthernet1/1
 Status         :Forwarding                 Status         :Forwarding

VLAN ID     : 1
Domain Name : Cisco MRP Ring 1
Domain ID   : FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF

Topology Change Request Interval        : 10ms
Topology Change Repeat Count            : 3
Short Test Frame Interval               : 10ms
Default Test Frame Interval             : 20ms
Test Monitoring Interval Count          : 3
Test Monitoring Extended Interval Count : N/A


Topology Change Request Interval        : 10ms
Topology Change Repeat Count            : 3
Short Test Frame Interval               : 10ms
Default Test Frame Interval             : 20ms
Test Monitoring Interval Count          : 3
Test Monitoring Extended Interval Count : N/A
```

Verifying the Configuration

You can use the following commands to verify the MRP configuration.

Command	Description
show mrp ring? {**1 - 22**}	Display details about the MRP ring configuration.
show mrp ports	Display details about the MRP port states. If MRP is not configured on any ports, display shows N/A.

Command	Description
show mrp ring {**1 - 22**} **statistics** [**all** | **event** | **hardware** | **packet** | **platform**]	Display details about the MRP ring operation.
debug mrp-ring [**alarm**|**cli** | **client** | **license** | **manager** | **packet** | **platform**]	Trace MRP events. **Note** **manager** is available only when the switch is configured as manager or automanager.
show tech-supportmrp	Display all MRP details.

CHAPTER 175

High-availability Seamless Redundancy

High-availability Seamless Redundancy

High-availability Seamless Redundancy overview

HSR is defined in International Standard IEC 62439-3-2016 clause 5. HSR is similar to Parallel Redundancy Protocol (PRP) but is designed to work in a ring topology. Instead of two parallel independent networks of any topology (LAN-A and LAN-B), HSR defines a ring with traffic in opposite directions. Port-A sends traffic counter clockwise in the ring, and Port-B sends traffic clockwise in the ring.

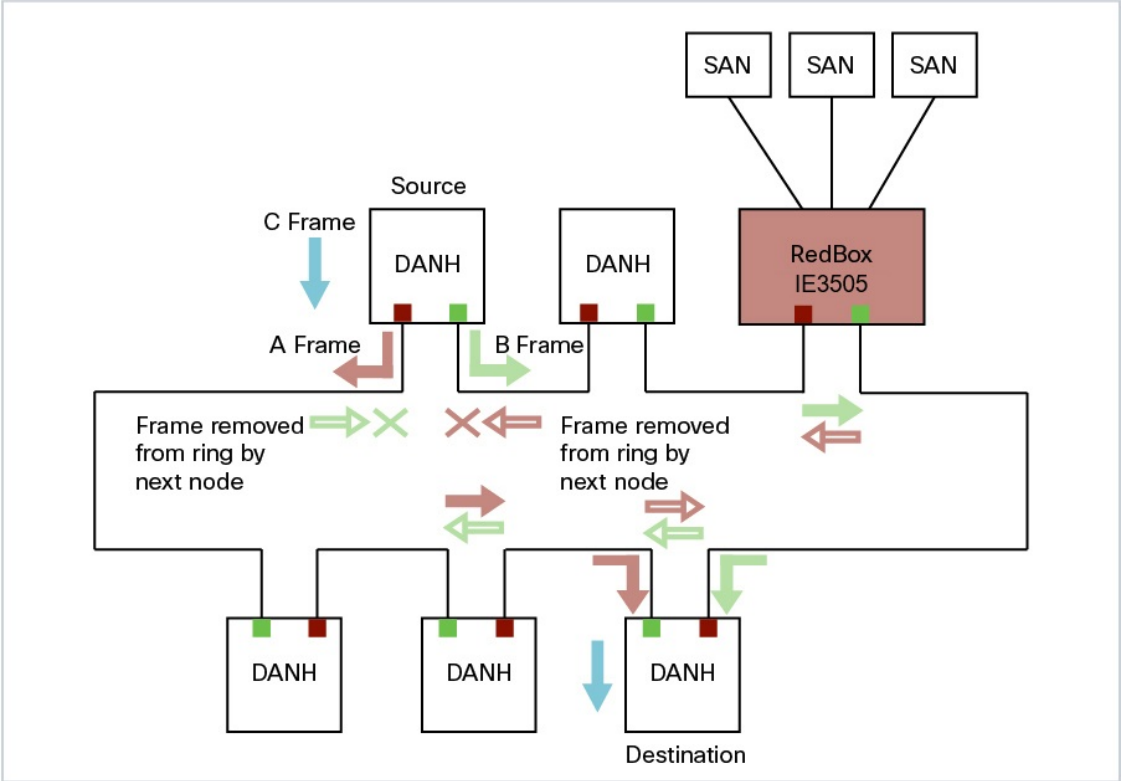
The HSR packet format is also different from PRP. To allow the switch to determine and discard duplicate packets, additional protocol specific information is sent with the data frame. For PRP, this is sent as part of a trailer called the redundancy control trailer (RCT), whereas for HSR this is sent as part of the header called the HSR header. Both the RCT and HSR header contain a sequence number, which is the primary data used to determine if the received frame is the first instance or a duplicate instance.

In this release, the switch supports only HSR-singly attached node (SAN) and only one HSR instance. If you have created a PRP instance, no HSR instance can be created.

The non-switching nodes with two interfaces attached to the HSR ring are referred to as Doubly Attached Nodes implementing HSR (DANHs). Similar to PRP, Singly Attached Nodes (SANs) are attached to the HSR ring through a device called a RedBox (Redundancy Box). The RedBox acts as a DANH for all traffic for which it is the source or the destination. The switch implements RedBox functionality using Gigabit Ethernet port connections to the HSR ring.

The following figure shows an example of an HSR ring as described in IEC 62439-3.

Figure 187: Example of HSR Ring Carrying Unicast Traffic



Devices that do not support HSR out of the box (for example, laptops and printers) cannot be attached to the HSR ring directly because all HSR capable devices must be able to process the HSR header on packets received from the ring and add the HSR header to all packets sent into the ring. These nodes are attached to the HSR ring through a RedBox. As shown in the figure above, the RedBox has two ports on the DANH side. Non-HSR SAN devices are attached to the upstream switch ports. The RedBox generates the supervision frames on behalf of these devices so that they are seen as DANH devices on the ring. Because the RedBox emulates these as DANH, they are called Virtual Doubly Attached Nodes (VDAN).

Switches that Support HSR

Table 189: The following Advanced base modules SKUs (PIDs) supports HSR.

Switch	PID
Cisco IE3505 Rugged Series Switch	IE-3505-8P3S
	IE-3505-8T3S
Cisco IE3505 Heavy-Duty Series Switch	IE-3505H-16T

Support for HSR is available on Network Essentials and Network Advantage licenses.

Supported HSR Features

IE-3505-8P3S, IE-3505-8T3S, and IE-3505H-16T switches support the following HSR features.

Maximum of one HSR ring is supported as shown in the following table:

Table 190: HSR Support for Cisco IE3505 Series Switch and Expansion Models

Switch	FPGA Profile	Number of Rings
Cisco IE3505 Rugged Series without expansion module	Default	1
	Redundancy	1
Cisco IE3505 Rugged Series with expansion module	Default	1
	Redundancy	1
IE-3505H-16T	Default	1
	Redundancy	1

Loop Avoidance

Each node in the HSR ring forwards frames received from one port to the other port of the HSR pair. To avoid loops and use network bandwidth effectively, the RedBox does not transmit frames that are already transmitted in same direction. When a node injects a packet into the ring, the packet is handled as follows to avoid loops:

- Unicast packet with destination inside the ring: When the unicast packet reaches the destination node, the packet is consumed by the respective node and is not forwarded.
- Unicast packet with destination not inside the ring: Because this packet does not have a destination node in the ring, it is forwarded by every node in the ring until it reaches the originating node. Because every node has a record of the packet it sent, along with the direction in which it was sent, the originating node detects that packet has completed the loop and drops the packet.
- Multicast packet: A multicast packet is forwarded by each node because there can be more than one consumer of this packet. For this reason a multicast packet always reaches the originating node. However, every node will check whether it has already forwarded the received packet through its outgoing interface. Once the packet reaches the originating node, the originating node determines that it already forwarded this packet and drops the packet instead of forwarding it again.

HSR RedBox Modes of Operation

The most basic mode of operation is HSR-SAN mode (single RedBox mode). In this mode, the RedBox is used to connect SAN devices to the HSR ring. The Redbox's responsibility in this mode is to represent SAN devices as VDANs on the ring.

HSR SAN Mode

In HSR-SAN mode, the RedBox inserts the HSR tag on behalf of the host and forwards the ring traffic, except for frames sent by the node itself, duplicate frames, and frames for which the node is the unique destination. In this mode, packets are handled as follows:

- A source DANH sends a frame passed from its upper layers (C frame), prefixes it with an HSR tag to identify frame duplicates, and sends the frame over each port (A frame and B frame).

- A destination DANH receives two identical frames from each port within a certain interval. The destination DANH removes the HSR tag of the first frame before passing it to its upper layers and discards any duplicate.
- Each node in the HSR ring forwards frames received from one port to the other port of the HSR pair. A node will not forward frames received on one port to the other under the following conditions:
  - The received frame returns to the originating node in the ring.
  - The frame is a unicast frame with a destination MAC address of a node upstream of the receiving node.
  - The node had already sent the same frame in the same direction. This rule prevents a frame from spinning in the ring in an infinite loop.

HSR-SAN interfaces

HSR-SAN mode is supported on interfaces GigabitEthernet 1/1-2 and GigabitEthernet 1/4-5. HSR ring 1 is configured as a pair of ports: Gi1/1 and Gi1/2 or Gi1/4 and Gi1/5.

Table 191: The supported HSR-SAN channel interfaces for IE3505 Rugged Series Switch

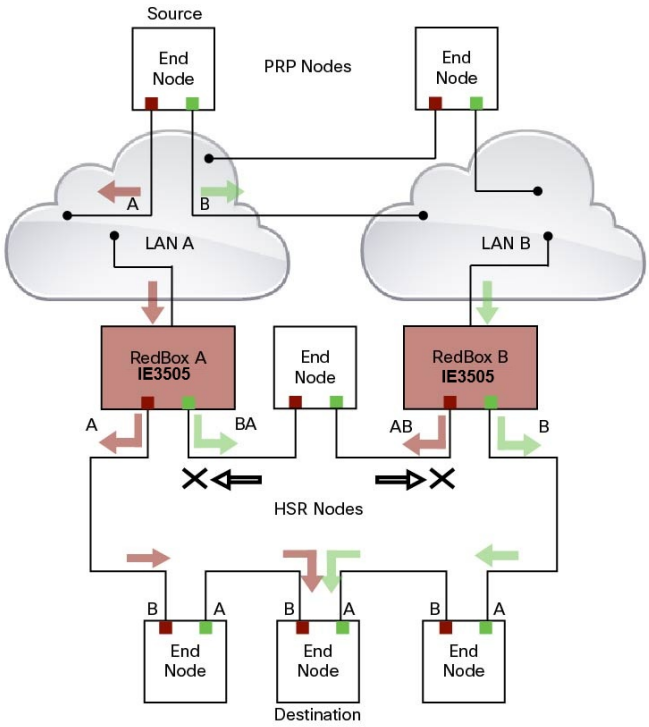
Mode	Module	Port Interface
HSR-SAN ring 1	Base Module	GigabitEthernet1/1 and 1/2 or GigabitEthernet1/4 and 1/5

HSR-PRP (Dual RedBox Mode)

HSR-PRP mode, also called Dual RedBox mode, is used to bridge HSR and PRP networks. Dual RedBox mode is supported on the Cisco IE3505 Series Switch.

In this mode, two different RedBoxes connect to LAN A and LAN B of the PRP network. Two ports connect to the HSR ring and one port connects to one of the two PRP LANs. The traffic on the upstream interlink port connecting the RedBox to the PRP network is PRP-tagged. In HSR-PRP mode, the RedBox extracts data from the PRP frame and generates the HSR frame using this data, and performs the reverse in the opposite direction. To avoid loops and use network bandwidth effectively, the RedBox does not transmit frames already transmitted in same direction (see Loop Avoidance, on page 2615).

The following figure shows an HSR ring connected to a PRP network through two RedBoxes, one for each LAN. In this example, the source frame originates in the PRP network. RedBoxes are configured to support PRP traffic on the interlink ports and HSR traffic on the ring ports. Nodes connected to the HSR-PRP Redbox act as a SAN to the PRP Redbox and a VDAN to the HSR-PRP Dual Redbox.

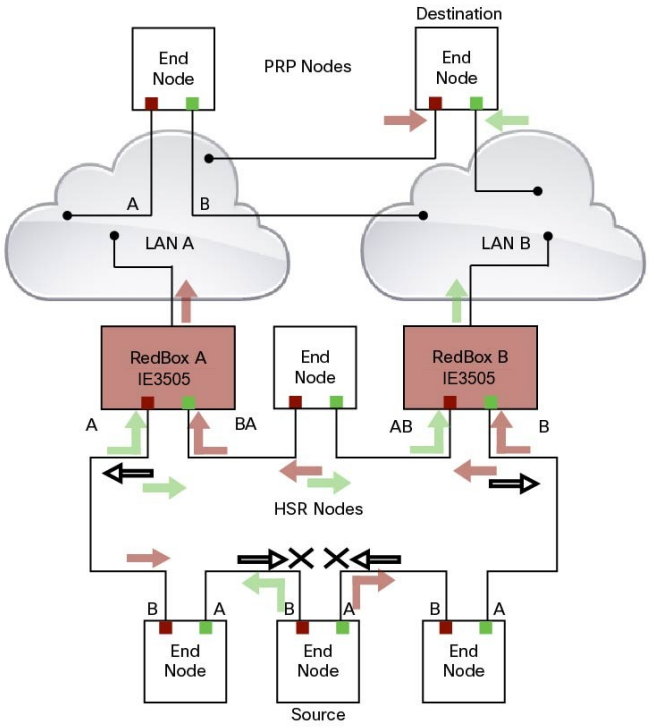


The sequence number from the PRP RCT is reused for the HSR tag and vice versa to allow frame identification from one redundancy network to the other and to identify the pairs and duplicates on either network. In the figure above, RedBox A and RedBox B send the same frame (A and AB and B and BA, respectively), but a RedBox does not transmit a frame that it already received.

Every DANH device generates its own sequence number, which is incremented for each outgoing frame. When a packet is switched from HSR to PRP or PRP to HSR, the sequence number is taken from the incoming packet so the same sequence number is used. Any node, whether it is an intermediate or final destination in the HSR or PRP network, uses the source MAC address and sequence number as the key for duplicate packet detection. Because the source address is expected to be unique for each node, there are no overlapping sequence numbers between different nodes.

Multicast frames or unicast frames without a receiver in the ring (arrows with the black outline in the figure) are removed by the RedBox that inserted them into the ring, if they originated from outside the ring. For this purpose, the frames carry a LAN identifier that is also the RedBox identifier.

The following figure shows an HSR ring coupled to a PRP network, where the source frame originates in the HSR ring.



To prevent frames from being reinjected into the PRP network through the other RedBox, each HSR frame carries the 4-bit PathId, which identifies the PRP network from which the frame came originally. RedBoxes are configured and identified by the PathId of the PRP LAN to which they are attached.

Different PathIds can be used to bridge more than one PRP network to an HSR ring. Likewise, more than one HSR ring can be bridged to a PRP network.

PRP is not needed for HSR-PRP to function in the IE3500 Rugged Series Switches. Any third port can be connected to PRP LAN A or LAN B network without PRP or any specific configurations.

PRP Supervision frames are sent toward PRP LAN A or LAN B from conversion of HSR Supervision Frames originated from DANHs and VDANs of HSR RedBoxes in the HSR ring. The HSR-PRP Redbox does not generate them but passes them along.

Packet flow in HSR-PRP

Packets coming from PRP network in the coupled PRP LAN-A or LAN-B are expected to have an RCT (Redundancy Control Trailer) tag. The switch removes the RCT and transfers the information to the HSR header using the programmed Net ID and LAN ID, recalculates the CRC, and sends the modified packet out to both ring A and ring B. If the packets originate from a SAN in the coupled PRP network, the switch treats it similarly as a VDAN to the HSR ring.

Egress Data Path—Packets coming from a SAN or PRP LAN A or LAN B to the HSR ring:

- For PRP packets, the switch converts the PRP RCT to an HSR tag for all packets (transfers Sequence Number and LAN ID from PRP to HSR).
- For SAN packets, the switch just inserts the HSR tag as is done in HSR-SAN RedBox mode.

- The switch needs to learn the MAC source address and add it to the Proxy Node table (VDAN table) with a new additional bit that allows the switch to distinguish between DANP or SAN. This allows the ingress path to determine whether to include the RCT trailer or not.

Ingress Data Path—Packets coming from the HSR ring to a SAN or PRP LAN A or LAN B:

- If the Proxy Node table or VDAN table lookup of the MAC destination address returns DANP, the switch converts the HSR tag to PRP RCT for accepted packets (transfers Sequence Number and LAN ID from HSR to PRP RCT).
- If the Proxy Node table or VDAN table lookup of the MAC destination address returns SAN, the switch strips the HSR tag and sends the packet without the RCT.

HSR-PRP Interfaces

In HSR-PRP Dual RedBox mode, two ports are connected to the HSR ring, and one port is connected to the PRP LAN A or LAN B network. The two ports that connect to the HSR ring are fixed (Gi1/1 and Gi1/2). When set to HSR-PRP mode, the two ports that connect to the HSR ring (Gi1/1 and Gi1/2) are automatically configured to HSR.

Table 192: The supported HSR-PRP channel interfaces for IE3505 Rugged Series Switch

Mode	Module	Port Interface
HSR-PRP	Base Module	GigabitEthernet1/1 and 1/2

The port connected to PRP LAN A or LAN B can be any other port from the base module or expansion module. All remaining ports of the HSR-PRP RedBox (base module or expansion module ports) can be used for any other purpose, for example, to connect a DHCP server.

These ports act as non-HSR/PRP nodes (SANs/VDANs) in the topology. The HSR-PRP RedBox can use all remaining ports (base module or expansion module ports) for other purposes, such as connecting a DHCP server. These non-PRP and non-HSR ports must be in the same VLAN as the HSR and PRP ports to achieve SAN/VDAN behavior.

Connecting Multiple PRP Networks to an HSR Ring

A maximum of six PRP networks, identified by the PathId, can be connected to the same HSR ring. The 4-bit PathId consists of the following:

- The 3-bit NetId (1 to 6), which identifies a PRP network and the two RedBoxes that connect the PRP network to an HSR ring.
- The 1-bit LanId (LAN A = 0 and LAN B = 1)

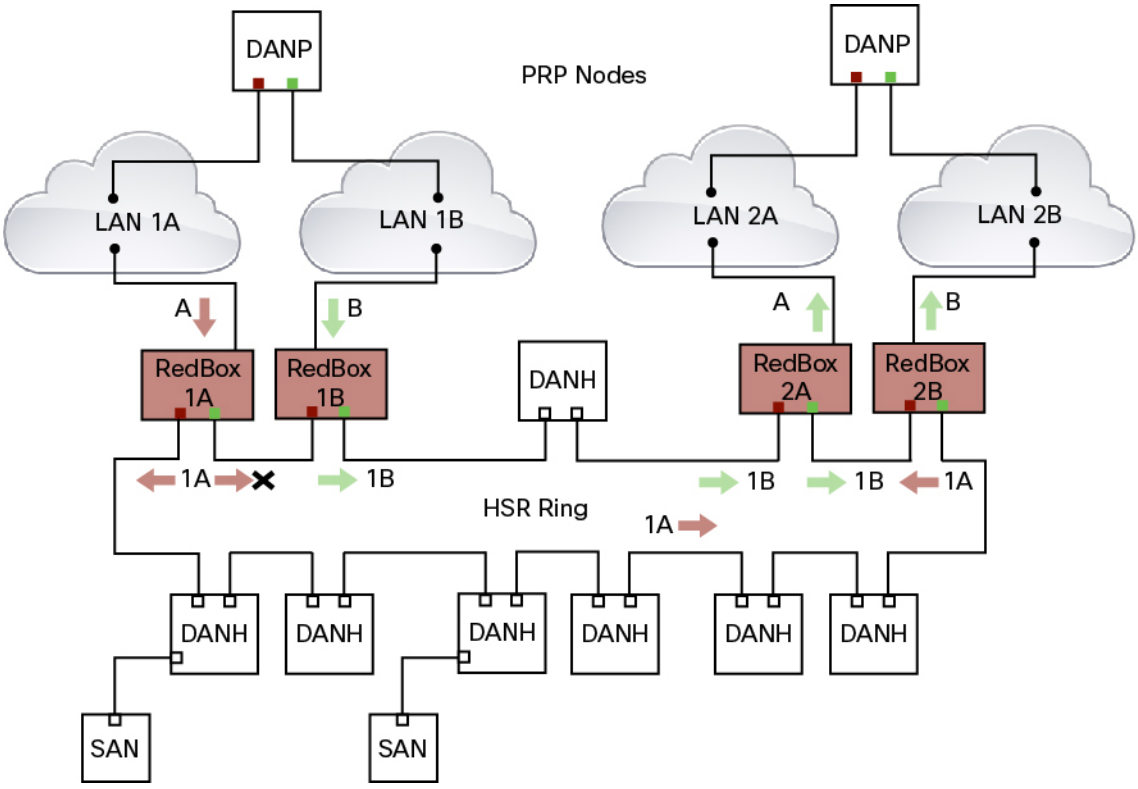
NetId values are as follows:

- 0 for regular HSR frames
- 1 to 6 for frames originating from a PRP network
- 7 is reserved

The following table lists the combinations of NetIds and LanIds for Redbox-A and Redbox-B.

PathId		
NetId	**LanId**	
	RedBox-A	**RedBox-B**
001	0	1
010	0	1
011	0	1
100	0	1
101	0	1
110	0	1
000	**Used for Local HSR Ring**	
111	**Reserved**	

The following figure shows an example of an HSR ring connected to two PRP networks.



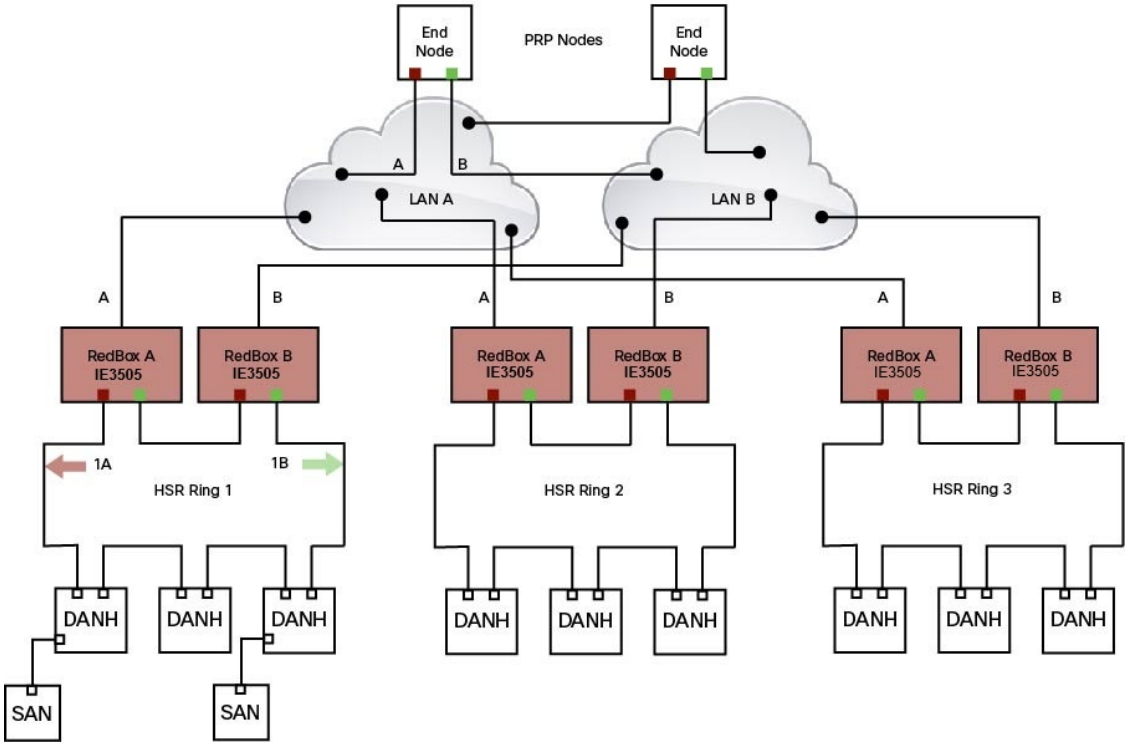
To prevent reinjection of frames coming from one PRP network into another PRP network or from the other LAN of the same PRP network, a RedBox only forwards frames that do not carry its own PathId from the HSR ring.

When a PRP frame from LAN A or from LAN B of a PRP network with a given NetId is inserted to the HSR ring, a RedBox inserts its own NetId and the LanId “A” or ”B” into the PathId of the HSR tag.

When forwarding a frame from the HSR ring to a PRP network, the RedBox inserts the LanId “A” or ”B” into the RCT.

Connecting Multiple HSR Rings to a PRP Network

A PRP network can be connected to any number of HSR rings, but these rings cannot be connected to each other because this would create loops. The following figure shows an example of three HSR rings connected to one PRP LAN.



CDP and LLDP for HSR

HSR supports the Cisco Discovery Protocol (CDP) and Link Layer Discovery Protocol (LLDP). CDP and LLDP are Layer 2 neighbor discovery protocols. Both CDP and LLDP can provide information about nodes directly connected to the device. They also provide additional information such as the local and remote interface and device names.

When CDP or LLDP is enabled, you can use the CDP or LLDP information to find the adjacent nodes on an HSR ring and their status. You can then use the neighbor information from each node to determine the complete HSR network topology and debug and locate ring faults.

CDP and LLDP are configured on physical interfaces only.

For more information, see Configuring an HSR Ring and Verifying Configuration.

PTP over HSR

Precision Time Protocol (PTP) is supported on the IE3500 Rugged and IE3500H Heavy Duty Series Switches for the PTP Power Profile only.

Because the PTP 1588 standard does not currently account for clocks synchronized over redundant, simultaneously active paths, HSR must handle PTP packets differently that other packet types. To provide high availability for PTP through redundancy, the HSR duplicate/discard logic is not used for PTP packets.

To understand how PTP clock syncronization works in an HSR network, suppose that a VDAN/SAN is the PTP grandmaster clock (GMC). Dually attached devices receive PTP synchronization information over both their HSR ports. However, only one of the ports (referred to as time recipient) is used to synchronize the local clock. The other HSR port (referred to as PASSIVE) continues to receive synchronization information, but is not used to synchronize the local clock. Suppose that RedBox 2 has its port-A as time recipient and port-B as PASSIVE. When port-A goes down, the port-B port takes over as the time recipient and is used to continue synchronizing the local clock on RedBox 2.

Note Cisco is moving from the traditional Master/Slave nomenclature. In this document, the terms *Grandmaster clock (GMC)* or *time source* and *time recipient* are used instead.

The PTP grandmaster in an HSR network can be a RedBox, a VDAN/SAN, or a DANH.

To use PTP over HSR, configure HSR and PTP separately. PTP over HSR works without any additional configuration. Note that in most cases, you do not need to perform any PTP configuration on the interfaces because PTP is enabled by default on all physical ethernet interfaces.

Supported PTP Profiles and Modes

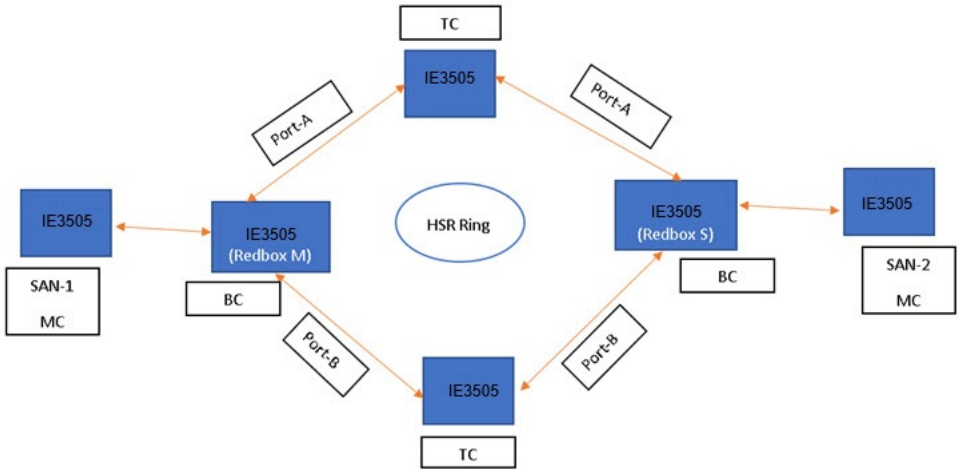
PTP over HSR is supported only for the for the PTP Power Profile. For unsupported PTP profiles, PTP traffic flows over HSR port-A only.

The following table shows the HSR support for PTP profiles, clock modes and RedBox types.

PTP Profile	Clock Mode	Supported?	HSR Redbox Type as per IEC 62439-3
Power Profile	BC	Yes	HSR RedBox as doubly attached BC (DABC) with P2P
	P2P TC	Yes	HSR RedBox as doubly attached TC (DATC) with P2P
	GMC-BC	No	Not applicable
	Forward	No	Not applicable
Default Profile	BC	No	Not applicable
	E2E TC	No	Not applicable

HSR RedBox as Doubly Attached BC (DABC) with P2P

This section describes the operation of PTP over HSR using an example where RedBox M and RedBox S are configured to run in Power Profile as Boundary Clocks that use the Peer-to-Peer delay measurement mechanism.



Assume for this example that SAN-1 is the GMC. All the clocks are configured to run Peer-to-Peer Delay measurement and the peer delay is regularly calculated and maintained on every link shown in the figure. The BMCA on RedBox M determines the port to SAN-1 to be connected to the time source. The PTP protocol running on RedBox M will forward Sync and Follow_up messages on ports A and B.

On RedBox S, the regular BMCA operation determines port A to be time recipient and port B to be PASSIVE. However, with the knowledge that ports A and B are part of the same HSR ring, port B is forced into PASSIVE_SLAVE state and port A becomes active for PTP.

Port A works as a regular time recipient port. It uses the Sync and Follow_Up messages along with their correction field to calculate the delay and offset from time source and synchronize the local clock. (Unlike an E2E BC, it does not need to generate Delay_Req messages since all the link delays and residence times along the PTP path are accumulated in the correction field of the Follow_Up messages.)

Port B, which is in PASSIVE_SLAVE state operates as follows: Just like port A, it maintains the delay and offset from time source, but does not perform any operation on the local clock. Having all the synchronization information available enables it to seamlessly take over as the new time recipient in case port A loses communication with the GMC. Note that on IE switch platforms we currently do not support PTP profile conversion. For example, if RedBox S in the figure above were an IE switch, it would not support the Delay_Req/Delay_Resp message exchange. It would only support the Peer-to-Peer delay measurement mechanism using PDelay messages.

Configuration Example

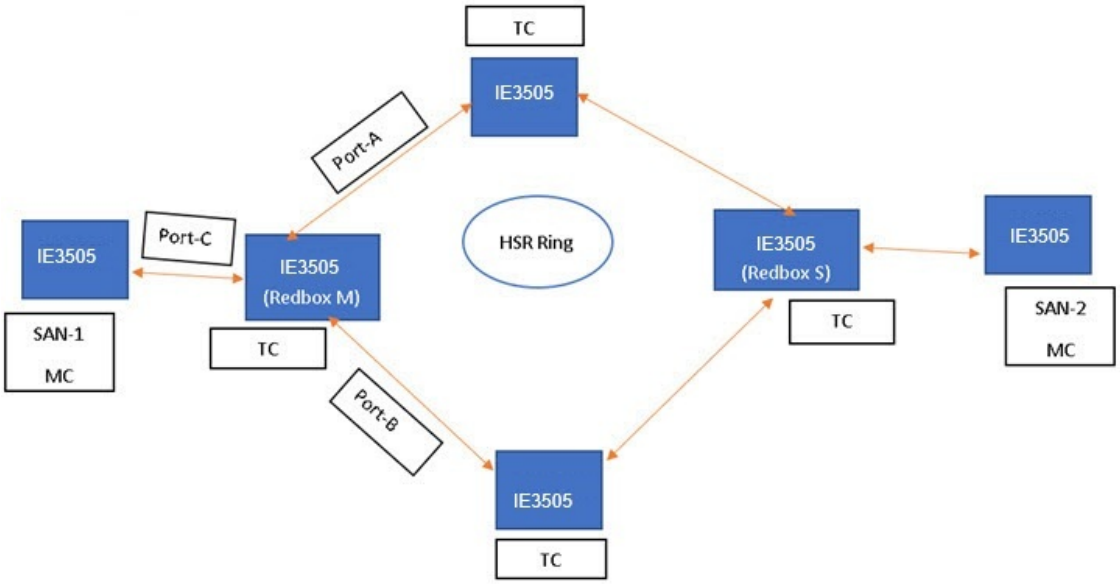
```
SAN-1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
SAN-1(config)#ptp profile power
SAN-1(config)#ptp mode boundary pdelay-req
SAN-1(config)#ptp priority1 1
SAN-1(config)#end

SAN-2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
SAN-2(config)#ptp profile power
SAN-2(config)#ptp mode boundary pdelay-req
```

```
SAN-2(config)#end

REDBOX-M#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
REDBOX-M(config)#ptp profile power
REDBOX-M(config)#ptp mode boundary pdelay-req
REDBOX-M(config)#end

REDBOX-S#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
REDBOX-S(config)#ptp profile power
REDBOX-S(config)#ptp mode boundary pdelay-req
REDBOX-S(config)#end

DANH-TOP#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
DANH-TOP(config)#ptp profile power
DANH-TOP(config)#ptp mode p2ptransparent
DANH-TOP(config)#end

DANH-BOTTOM#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
DANH-BOTTOM(config)#ptp profile power
DANH-BOTTOM(config)#ptp mode p2ptransparent
DANH-BOTTOM(config)#end

SAN-1#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Parent Port Number: 0
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128

SAN-2#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:29:C2:FF:FE:3C:6A:C0
  Parent Port Number: 9
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128

REDBOX-M#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
```

```
  Parent Port Number: 3
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128

REDBOX-S#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:29:C2:FF:FE:3C:5D:80
  Parent Port Number: 3
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128

DANH-TOP#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:29:C2:FF:FE:3C:5D:80
  Parent Port Number: 3
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128

DANH-BOTTOM#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:29:C2:FF:FE:3C:5D:80
  Parent Port Number: 4
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128
```

HSR RedBox as Doubly Attached TC (DATC) with P2P

This section describes the operation of PTP over HSR using an example where RedBox M and RedBox S are configured to run in Power Profile as Transparent Clocks.



Assume for this example that SAN-1 is the GMC. All the clocks are configured to run Peer-to-Peer Delay measurement and the peer delay is regularly calculated and maintained on every link shown in the figure. RedBox M and RedBox S run BMCA even though it is not mandatory for a P2P TC to run BMCA. On RedBox M, the BMCA on redbox M determines the port to SAN-1 to be connected to the time source. RedBox M forwards all Sync and Follow_Up messages received on port C out of ports A and B.

On RedBox S, port A is determined to be time recipient and port B to be PASSIVE_SLAVE as described earlier.

Port A operates as follows: It uses the Sync and Follow_Up messages along with their correction field to calculate the delay and offset from time source and synchronize the local clock. (Unlike a E2E BC, it does not need to generate Delay_Req messages since all the link delays and residence times along the PTP path are accumulated in the correction field of the Follow_Up messages.) It also forwards the Sync and Follow_Up messages out of port C.

Port B operates as follows: Just like port A, it maintains the delay and offset from time source, but does not perform any operation on the local clock. Having all the synchronization information available enables it to seamlessly take over as the new time recipient in case port A loses communication with the GMC. Post-processing, it drops the Sync/Follow_Up messages since the copy of Sync/Follow_Up that arrives on port A is forwarded out of port C.

Configuration Example

```
SAN-1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
SAN-1(config)#ptp profile power
SAN-1(config)#ptp mode boundary pdelay-req
SAN-1(config)#ptp priority1 1
SAN-1(config)#end
SAN-2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
```

```
SAN-2(config)#ptp profile power
SAN-2(config)#ptp mode boundary pdelay-req
SAN-2(config)#end
REDBOX-M#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
REDBOX-M(config)#ptp profile power
REDBOX-M(config)# ptp mode p2ptransparent
REDBOX-M(config)#end
REDBOX-S#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
REDBOX-S(config)#ptp profile power
REDBOX-S(config)# ptp mode p2ptransparent
REDBOX-S(config)#end
DANH-TOP#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
DANH-TOP(config)#ptp profile power
DANH-TOP(config)#ptp mode p2ptransparent
DANH-TOP(config)#end
DANH-BOTTOM#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
DANH-BOTTOM(config)#ptp profile power
DANH-BOTTOM(config)#ptp mode p2ptransparent
DANH-BOTTOM(config)#end
SAN-1#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Parent Port Number: 0
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128
SAN-2#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Parent Port Number: 3
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128
REDBOX-M#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Parent Port Number: 3
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
```

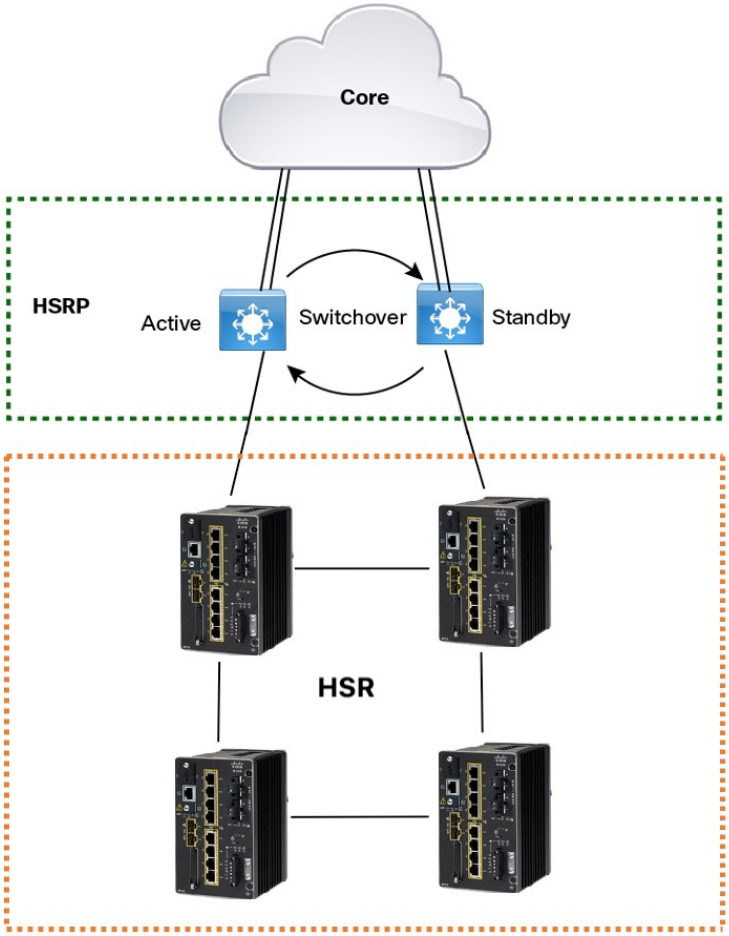
```
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128
REDBOX-S#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Parent Port Number: 3
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128
DANH-TOP#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Parent Port Number: 3
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128
DANH-BOTTOM#sh ptp parent
 PTP PARENT PROPERTIES
  Parent Clock:
  Parent Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Parent Port Number: 3
  Observed Parent Offset (log variance): N/A
  Observed Parent Clock Phase Change Rate: N/A

  Grandmaster Clock:
  Grandmaster Clock Identity: 0x0:35:1A:FF:FE:94:4F:0
  Grandmaster Clock Quality:
 Class: 248
 Accuracy: Unknown
 Offset (log variance): N/A
 Priority1: 1
 Priority2: 128
```

HSR Uplink Redundancy Enhancement

The HSR Uplink Redundancy Enhancement feature allows for flexible designs that enable two separate interfaces to connect upstream from the HSR ring through two separate HSR RedBoxes. This ensures there is no single point of failure exiting the HSR ring. Examples of protocols that can leverage this feature to improve high availability include HSRP, VRRP and REP. Prior to this enhancement, if these protocols were

utilized on redundant uplinks, undesirable results could occur, such as next-hop split-brain conditions or slow REP failover times.

The following diagram shows an example network with HSR and HSRP that allows uplink next-hop gateway redundancy out of the HSR ring.



To implement HSR Uplink Redundancy, ensure that the **fpgamode-DualUplinkEnhancement** feature is not disabled. This feature is required to support the connectivity to a dual router (HSRP in this case) on the distribution layer:

```
Switch#show hsr ring 1 detail | include fpgamode
 fpgamode-DualUplinkEnhancement: Enabled
```

If the output shows *fpgamode-DualUplinkEnhancement,:Disabled* issue the following command:

```
Switch# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)# hsr-ring 1 fpgamode-DualUplinkEnhancement
Switch(config)# end
```

HSRP Configuration

The following example HSRP configuration applies to the two distribution switches Active & Standby in the above figure. In the following configuration, HSRP is configured in a Switch Virtual Interface (SVI).

```
Active# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Active(config)# interface vlan 10
Active(config-if)# ip address 30.30.30.2 255.255.255.0
Active(config-if)# standby 1 ip 30.30.30.1
Active(config-if)# standby 1 priority 120
Active(config-if)# end

Standby# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Standby(config)# interface Vlan10
Standby(config-if)# ip address 30.30.30.4 255.255.255.0
Standby(config-if)# standby 1 ip 30.30.30.1
Standby(config-if)# end

Active# show standby
Vlan10 - Group 1
  State is Active
    8 state changes, last state change 00:03:55
    Track object 1 (unknown)
  Virtual IP address is 30.30.30.1
  Active virtual MAC address is 0000.0c07.ac01 (MAC In Use)
    Local virtual MAC address is 0000.0c07.ac01 (v1 default)
  Hello time 200 msec, hold time 750 msec
    Next hello sent in 0.176 secs
  Preemption enabled, delay min 5 secs, reload 5 secs, sync 5 secs
  Active router is local
  Standby router is 30.30.30.4, priority 100 (expires in 0.656 sec)
  Priority 120 (configured 120)
  Group name is "hsrp-Vl10-1" (default)
  FLAGS: 0/1
Active# show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P State   Active          Standby         Virtual IP
Vl10        1    120 P Active  local           30.30.30.4      30.30.30.1


Standby# show standby
Vlan10 - Group 1
  State is Standby
    13 state changes, last state change 00:04:17
    Track object 1 (unknown)
  Virtual IP address is 30.30.30.1
  Active virtual MAC address is 0000.0c07.ac01 (MAC Not In Use)
    Local virtual MAC address is 0000.0c07.ac01 (v1 default)
  Hello time 200 msec, hold time 750 msec
    Next hello sent in 0.064 secs
  Preemption enabled, delay min 5 secs, reload 5 secs, sync 5 secs
  Active router is 30.30.30.2, priority 120 (expires in 0.816 sec)
  Standby router is local
  Priority 100 (default 100)
  Group name is "hsrp-Vl10-1" (default)
  FLAGS: 0/1
Standby# show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P State   Active          Standby         Virtual IP
Vl10        1    100 P Standby 30.30.30.2      local           30.30.30.1
```

Guidelines and Limitations

- HSR is supported only in a standalone deployment.
- Only one HSR instance is supported. Note that the switch supports only one HSR or one PRP instance, so if a PRP instance has been created, no HSR instance can be created.
- HSR ring 1 can only be configured as a pair of ports: Gi1/1 and Gi1/2 or Gi1/4 and Gi1/5. Using these port pairs, you can configure 1 HSR ring.
- The HSR feature requires the Network Essentials license.
- The HSR feature is not enabled by default and you must explicitly configure the HSR rings.
- HSR is disabled automatically if the required firmware image is not available on the system.
- Once a port is part of a ring, the media-type, speed, and duplex settings of the port cannot be changed. We recommend that you apply those settings before configuring ring membership.
- If mode of HSR interfaces is changed from access to trunk mode or vice-versa after configuring the ring,we recommended that you flap the HSR ring.
- The recommended maximum number of nodes in the node table is 512. Nodes are all the DANH and VDAN devices that can be connected to the ring at same time. This number is not an absolute limit, but higher numbers of entries may increase the number of duplicate packets received by the end devices.
- HSR ring ports can only be configured in L2 mode.
- HSR is supported on following port types:
    - 100 mbps, Full Duplex. Half duplex is not supported.
    - 1000 mbps, Full Duplex. Half duplex is not supported.
    - HSR is not supported on the uplink ports.
- Both ports of one ring must be of same speed and type (that is, both can be SFPs or both can be copper)
- The following protocols and features are mutually exclusive with HSR on the same port:
    - PRP
    - EtherChannels
    - Link Aggregation Control Protocol (LACP)
    - Port Aggregation Protocol (PAgP)
    - Resilient Ethernet Protocol (REP)
- MACsec, HSR, and PRP are not allowed together.
- HSR supports an MTU size of up to 1998 bytes of Ethernet payload.
- STP is not supported on the HSR ring. By default, all modes of Spanning Tree Protocol (STP) will be disabled on the ring ports.

- Switched Port Analyzer (SPAN) and Remote SPAN (RSPAN) are not supported on HSR. That is, SPAN and RSPAN should not be used to monitor the traffic on an HSR ring. In addition, traffic that has been monitored using RSPAN should not be transferred over an HSR ring.
- It is important for all interfaces in an HSR ring to have the same speed and duplex settings. It is recommended to apply those settings before configuring ring membership.
- Once a port is part of ring, the port cannot be shut down.

  For example, if Gi1/4 and Gi1/5 are part of an HSR ring and you try to shut down Gi1/4 or Gi1/5, the operation will not be permitted:

```
Switch(config)# interface gi1/4
Switch(config-if)#shutdown
 %Interface GigabitEthernet1/4 is configured in a HSR ring shutdown not permitted!
Switch(config-if)#
```

  You can perform a shutdown of the HSR ring. For example:

```
Switch# conf t
Switch(config)#int hs1
Switch(config-if)#shut
```

- VLAN configuration such as trunk and access mode must be the same on both the ports participating in the ring. For example, if Gi1/4 and Gi1/5 in an HSR ring are in trunk mode and you attempt to change the mode of one port to access, the ports in the ring will not be bundled:

```
Switch(config)#interface range gi1/4
Switch(config-if)#switchport mode  access
Jul 27 22:00:27.809 IST: %EC-5-CANNOT_BUNDLE2: Gi1/4 is not compatible with Gi1/5 and
will be suspended (trunk mode of Gi1/4 is access, Gi1/5 is dynamic)
```

- After an interface is added in the HSR ring, only the primary interface counters are updated. You should not need to configure and check the status of individual physical interfaces after they are added to the HSR ring.
- As soon as you configure an HSR ring on two ports of a switch, MAC flaps will be observed on other switches where the HSR configuration is yet to be applied. We recommend that you shut down the newly created HSR ring on the switch before configuring the ring on all switches, and then re-enable them one by one as shown below. For example, if there are four switches in the ring, disable the HSR ring interfaces on each switch:

```
Switch1(config)#interface range gi1/1-2
Switch1(config-if)#shutdown
Switch1(config-if)#hsr-ring hs1
Creating a HSR-ring interface hs1
Switch1(config-if)#int hs1
Switch1(config-if)#shutdown
Switch1(config-if)#end
```

  After all four switches are configured with the ring, re-enable the HSR ports on each switch:

```
Switch1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Switch1(config)#int hs1
Switch1(config-if)#no shutdown
Switch1(config-if)#end
Switch1#
```

  This prevents interim MAC flapping during HSR ring configuration in member switches.

Default Settings

Table 193: HSR Ring Parameters

Parameter	Description	Range	Default Value
entryForgetTime	Time for clearing an inactive entry from duplicate discard table.	0-65535	400 ms
fpgamode-DualUplinkEnhancement	Set FPGA register for source mac filtering.	enable or disable	enable
nodeForgetTime	Time to clear an inactive entry from the node table.	0-65535	60000 ms
nodeRebootInterval	Time after which the RedBox must start sending supervision frames after bootup.	0-65535	500 ms
pauseFrameTime	Time interval between HSR pause frames.	0-65535	25 ms
proxyNodeTableForgetTime	Time to clear an inactive entry from the proxy node table or vdan table.	0-65535	60000 ms
supervisionFrameLifeCheckInterval	Life check interval value for supervision frames.	0-65535	2000 ms
supervisionFrameOption			
mac-da	The last bytes of the destination MAC address of supervision frames (01:15:4E:00:01:00). The last 00 is replaced by the value of this parameter.	1-255 MAC DA last eight bits option value	No default
vlan-cfi	Enable Canonical Format Indicator (CFI) for the VLAN tagged frame.	enable or disable	disable
vlan-cos	Class of Service (COS) value to be set in the VLAN tag of the Supervision frame.	0-7	0
vlan-id	The VLAN tag of the supervision frame.	0-4095	0

Parameter	Description	Range	Default Value
vlan-tagged	Set VLAN tagging option.	enable or disable	disable
supervisionFrameRedboxMacaddress	The RedBox MAC address in the supervision frames.	48-bit RedBox MAC address	The interface HSR ring MAC address
supervisionFrameTime	Time interval between supervision frames.	0-65535	3 ms

Configure an HSR Ring

Follow these steps to configure an HSR ring:

Before you begin

- Read and understand the Guidelines and Limitations, on page 2631 section of this chapter.
- Ensure that the member interfaces of a HSR ring are not participating in any redundancy protocols such as FlexLinks, EtherChannel, REP, and so on before configuring a HSR ring.

Procedure

Step 1 Enter global configuration mode:

```
Switch# configure terminal
```

Step 2 (Optional) Globally enable CDP to provide information about HSR ring nodes:

```
Switch(config)# cdp run
```

Step 3 (Optional) Globally enable LLDP to provide information about HSR ring nodes:

```
Switch(config)# lldp run
```

Step 4 Enter interface configuration mode and disable PTP on the ports to be assigned to the HSR ring:

```
Switch(config)# interface range gi1/1-2
Switch(config-if-range)# no ptp enable
```

Step 5 (Optional) Enable CDP on the ports to be assigned to the HSR ring:

```
Switch(config-if-range)#cdp enable
```

Step 6 (Optional) Enable LLDP on the ports to be assigned to the HSR ring:

```
Switch(config-if-range)#lldp transmit
Switch(config-if-range)#lldp receive
```

Step 7 Shut down the ports before configuring the HSR ring:

```
Switch(config-if-range)# shutdown
```

Step 8 Create the HSR ring interface and assign the ports to the HSR ring:

```
Switch(config)# interface range gigabitEthernet 1/1-2
Switch(config-if-range)# hsr-ring 1
```

Step 9 (Optional) If required, configure HSR ring optional parameters. See the Default Settings section for the parameter descriptions, ranges and default values.

```
Switch(config-if-range)# hsr 1 supervisionFrameLifeCheckInterval 10000
```

Step 10 Turn on the HSR interface:

```
Switch(config-if-range)# no shutdown
Switch(config-if)# end
```

Example

```
Switch# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)# interface range gigabitEthernet 1/1-2
Switch(config-if-range)# no ptp enable
Switch(config-if-range)# shutdown
Switch(config-if-range)# hsr-ring 1
Switch(config-if-range)# hsr-ring 1 supervisionFrameLifeCheckInterval 10000
Switch(config-if-range)# no shutdown
Switch(config-if-range)# end
```

Configuring HSR-PRP

Follow these steps to enable HSR-PRP Redbox mode on the switch. Enabling HSR-PRP mode creates an HSR ring and bridges the the HSR ring to a PRP network.

Before you begin

- See Guidelines and Limitations, on page 2631.

Procedure

Step 1 Enter global configuration mode:

```
switch# configure terminal
```

Step 2 Enable HSR-PRP mode and select LAN A or LAN B and the optional PRP Net ID:

hsr-prp-mode enable {prp-lan-a | prp-lan-b} [1-6]

- prp-lan-a: Redbox is connected to LAN A.
- prp-lan-b: Redbox is connected to LAN B.
- 1-6: PRP Net ID value from 1 to 6.

  The default is 1.

Note
Be sure to configure the same Net ID in Redbox A and B that is part of the same PRP network.

Example:

```
switch(config)#hsr-prp-mode enable prp-lan-a
```

To disable HSR-PRP Redbox mode, use the command **no hsr-prp-mode enable**.

Clear All Node Table and VDAN Table Dynamic Entries

Procedure

Step 1 To clear all dynamic entries in the node table, enter the following command: **clear hsr node-table**

Step 2 To clear all dynamic entries in the VDAN table, enter the following command; **clear hsr vdan-table**

Verifying the Configuration

Command	Purpose
show hsr ring 1 [**detail**]	Displays configuration details for the specified HSR ring.
show hsr statistics {**egressPacketStatistics** | **ingressPacketStatistics** | **nodeTableStatistics** | **pauseFrameStatistics**}	Displays statistics for HSR components. **Note** To clear HSR statistics information, enter the command **clear hsr statistics**.
show hsr node-table	Displays HSR node table.
show hsr vdan-table	Displays HSR Virtual Doubly Attached Node (VDAN) table. **Note** The VDAN table and Proxy node table are the same.
show cdp neighbors	Displays CDP neighbor information for an HSR ring.
show lldp neighbors	Displays LLDP neighbor information for an HSR ring.

Configuration Examples

HSR-SAN

This example shows the configuration of an HSR ring (Ring 1) using Gi1/1 and Gi1/2 ports between four devices.

```
Switch-1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch-1(config)#interface range gigabitEthernet 1/1-2
Switch-1(config-if-range)#shutdown
Switch-1(config-if-range)#hsr-ring 1
Creating a HSR-ring interface HSR-ring 1

Switch-1(config-if-range)#no shutdown
Switch-1(config-if-range)#exit
Switch-1(config)#exit
Switch-1#
Switch-2# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch-2(config)#interface range gigabitEthernet 1/1-2
Switch-2(config-if-range)#shutdown
Switch-2(config-if-range)#hsr-ring 1
Creating a HSR-ring interface HSR-ring 1

Switch-2(config-if-range)#no shutdown
Switch-2(config-if-range)#exit
Switch-2(config)#exit
Switch-2#
Switch-3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch-3(config)#interface range gigabitEthernet 1/1-2
Switch-3(config-if-range)#shutdown
Switch-3(config-if-range)#hsr-ring 1
Creating a HSR-ring interface HSR-ring 1

Switch-3(config-if-range)#no shutdown
Switch-3(config-if-range)#exit
Switch-3(config)#exit
Switch-3#
Switch-4# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch-4(config)#interface range gigabitEthernet 1/1-2
Switch-4(config-if-range)#shutdown
Switch-4(config-if-range)#hsr-ring 1
Creating a HSR-ring interface HSR-ring 1

Switch-4(config-if-range)#no shutdown
Switch-4(config-if-range)#exit
Switch-4(config)#exit
Switch-4#
Switch-1# show hsr ring 1 detail
HSR-ring: HS1
------------
 Layer type = L2
 Operation Mode = mode-H
 Ports: 2       Maxports = 2
 Port state = hsr-ring is In use
 Protocol = Enabled  Redbox Mode = hsr-san
Ports in the ring:
```

```
  1) Port: Gi1/1
   Logical slot/port = 1/1      Port state = In use
        Protocol = Enabled
  2) Port: Gi1/2
   Logical slot/port = 1/2      Port state = In use
        Protocol = Enabled

Ring Parameters:
 Redbox MacAddr: 9433.d845.2a81
 Node Forget Time: 60000 ms
 Node Reboot Interval: 500 ms
 Entry Forget Time: 400 ms
 Proxy Node Forget Time: 60000 ms
 Supervision Frame COS option: 0
 Supervision Frame CFI option: 0
 Supervision Frame VLAN Tag option: Disabled
 Supervision Frame MacDa: 0x00
 Supervision Frame VLAN id: 0
 Supervision Frame Time: 3 ms
 Life Check Interval: 1600 ms
 Pause Time: 25 ms
 fpgamode-DualUplinkEnhancement: Enabled
```

CHAPTER 176

Device Level Ring

Device Level Ring

Device Level Ring overview

Device Level Ring (DLR) is a Layer 2 protocol that enables redundancy in a ring topology, providing fast network fault detection and reconfiguration for industrial networks. DLR is an EtherNet/IP™ protocol that is defined by the Open DeviceNet® Vendors' Association (ODVA).

DLR network includes at least one node configured to be a ring supervisor, and any number of normal ring nodes. All DLR ring nodes are required to have at least two Ethernet ports and incorporate embedded switch technology. Non-DLR multiport devices—switches or end devices—may be present in the ring, subject to certain implementation constraints. (No MAC table filtering is one example.) Non-DLR devices also affect the worst-case ring recovery time.

The DLR protocol supports a simple, single-ring topology. However, a network installation may use more than one DLR-based ring, so long as each ring is isolated so that DLR protocol messages from one ring are not present on another ring.

DLR supports redundant gateways for connecting with network infrastructure outside of the DLR network. The DLR redundant gateway feature provides mechanisms for automatically or manually selecting an active gateway. It also provides for automatic switchover to a backup gateway in the event of a connection failure.

A DLR ring can operate on access or trunk interfaces. A DLR ring configured with access ports can connect switches or end nodes. A DLR ring with trunk interfaces serves as an infrastructure that connects DLR-capable switches and devices in multiple VLANs. All the interfaces on the ring should have the same VLAN membership.

Components of DLR

DLR Device Classes

DLR supports the given classes of devices:

- *Ring supervisor*: On every DLR network, you must configure at least one device as the ring supervisor. The ring supervisor verifies the integrity of the ring, reconfigures it when a fault occurs, and collects diagnostic information. The ring supervisor also sends and processes Beacon frames at the default beacon interval of 400 microseconds.

  We recommend that you make at least one other device on the DLR network available as a back-up ring supervisor. Each supervisor is configured with a precedence value; the device with the highest precedence value becomes the active ring supervisor.

- *Beacon-based ring node*: This class of device implements the DLR protocol, but lacks ring supervisor capability. The device must be able to process and act on the beacon frames that the ring supervisor sends.

- *Redundant Gateway*: In a DLR network, redundant gateway devices enable multiple connections to the network outside of the DLR network. They provide an alternate path for communication in case a gateway device or its connection to the outside network fails.

  For information about redundant gateways, see the sections Redundant Gateways, on page 2645 and Configure a Redundant Gateway, on page 2654.

- *VLAN trunking*: VLAN trunking devices allow a DLR network to carry traffic through a trunk link. A trunk link connects switches that carry traffic from multiple VLANs, unlike an access link that can only carry a single VLAN.

  DLR VLAN trunking allows switches and star-connected devices in multiple VLANs to communicate through a DLR network. Traffic that passes from one switch to the next can either remain on the same VLAN or pass to a different VLAN through routing.

- *Multiple Rings support*: Cisco IE3500 Rugged Series Switches and Cisco IE3500 Heavy-Duty Series Switches support up to three rings.

Default and Redundancy FPGA Profiles

Because the DLR feature requires use of the Cisco IE3505 Series Switch FPGA, the number of DLR rings supported on the Cisco IE3505 Series Switch depends on the FPGA profile. The default FPGA profile on the Cisco IE3505 Series Switch base system supports three DLR rings, while the redundancy profile supports two DLR rings, regardless of the presence of an expansion module. For information about FPGA profile, see Configure FPGA Profile, on page 2239

The Cisco IE3505 Series Switch expansion modules have product ID prefix *IEM-3500*. The switch allow you to configure all DLR rings either on the base module, on the expansion module, or as a mix of both, without restrictions.

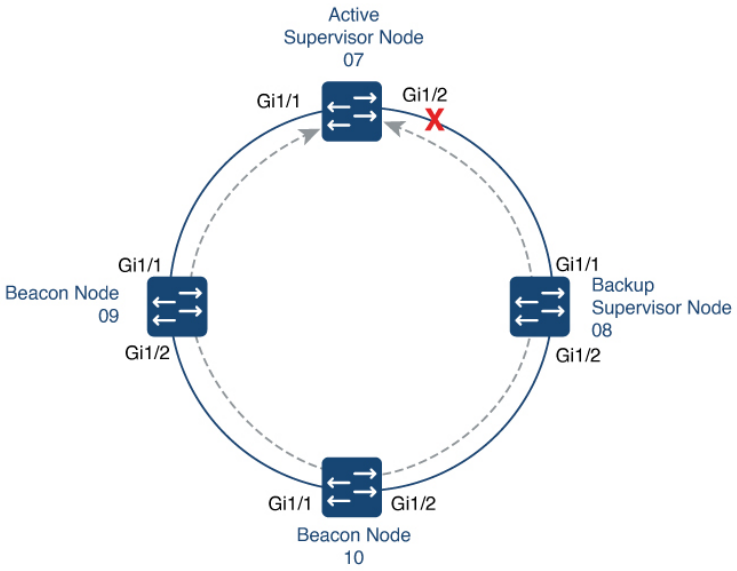
The Cisco IE3505 Series Switch can support a maximum of three DLR rings with the default FPGA profile and two rings with the redundancy profile.

DLR Topology

The Cisco IE3505 Rugged Series Switches and Cisco IE3505 Heavy-Duty Series Switches can act as a DLR ring supervisor, backup supervisor, or regular DLR beacon node. This functionality helps other nodes that are connected in a DLR with Cisco IE3505 Rugged Series Switches and Cisco IE3505 Heavy-Duty Series Switches to recover from a ring fault within 3 milliseconds and resume communications.

The following illustration shows a DLR ring with Cisco IE3505 Rugged Series Switches and Cisco IE3505 Heavy-Duty Series Switches acting as the ring supervisor, backup supervisor, and beacon nodes. The solid blue line represents the ring over which Ethernet frames travel, and the dotted gray line represents the bidirectional beacon frames. The X in the illustration shows where the ring supervisor blocks an interface to prevent broadcast storms. If a failure occurs in the DLR ring, the supervisor will unblock the interface.

Figure 188: DLR Topology



We recommend that you connect the interface with the higher number on the active supervisor node to the backup supervisor node.

Multiple Rings

The Cisco IE3505 Rugged Series Switches and IE3505 Heavy-Duty Series Switches support up to three rings.

Multiple Rings, Single Switch, Single VLAN

The following restrictions apply to multiple rings that are connected to one switch on one VLAN:

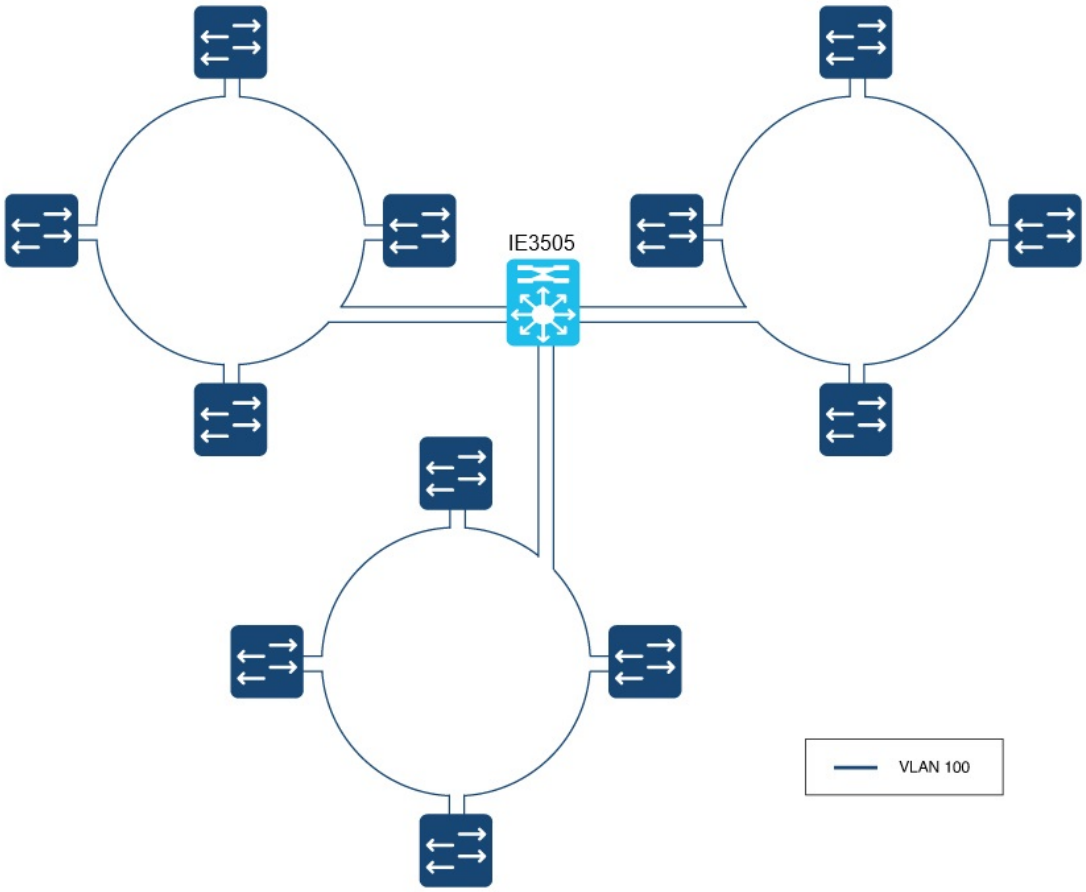
- Multiple rings cannot share the same ring ports.
- Ring ports supports access ports and trunk ports.

Note Ensure that all nodes in the same ring use the same access ports and trunk ports.

- All ports participating in the ring must have the same VLAN mode. If an access ring is configured, then all ports must be in the same access VLAN. For a trunk ring, all ports must be in trunk mode.

When only one node is a member of multiple rings, as in the example below, a VLAN can have ports in more than one ring.

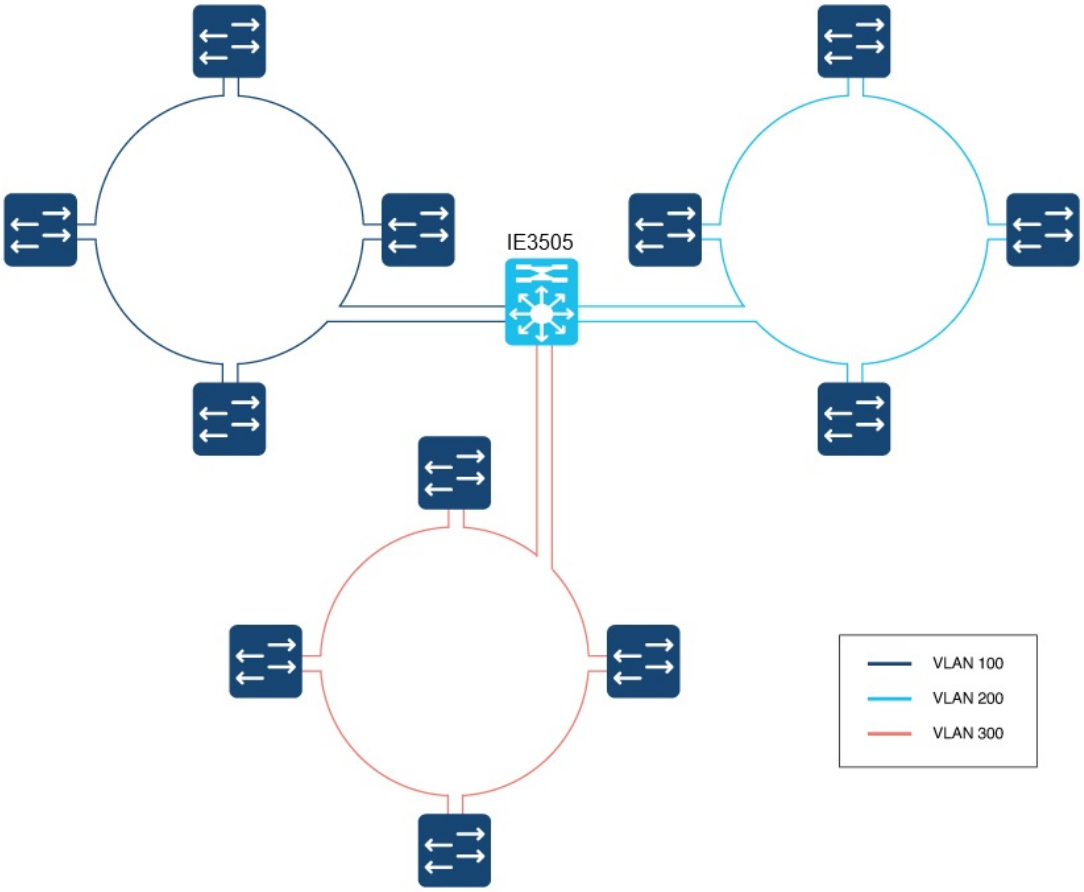
Figure 189: Multiple Rings, Single Switch, Single VLAN



Multiple Rings, Single Switch, Multiple VLANs

The following illustration shows multiple rings sharing a common supervisor with unique VLANs for each ring. When each DLR ring operates in a different VLAN, there is no issue and this is a supported deployment.

Figure 190: Multiple Rings, Single Switch, Multiple VLANs



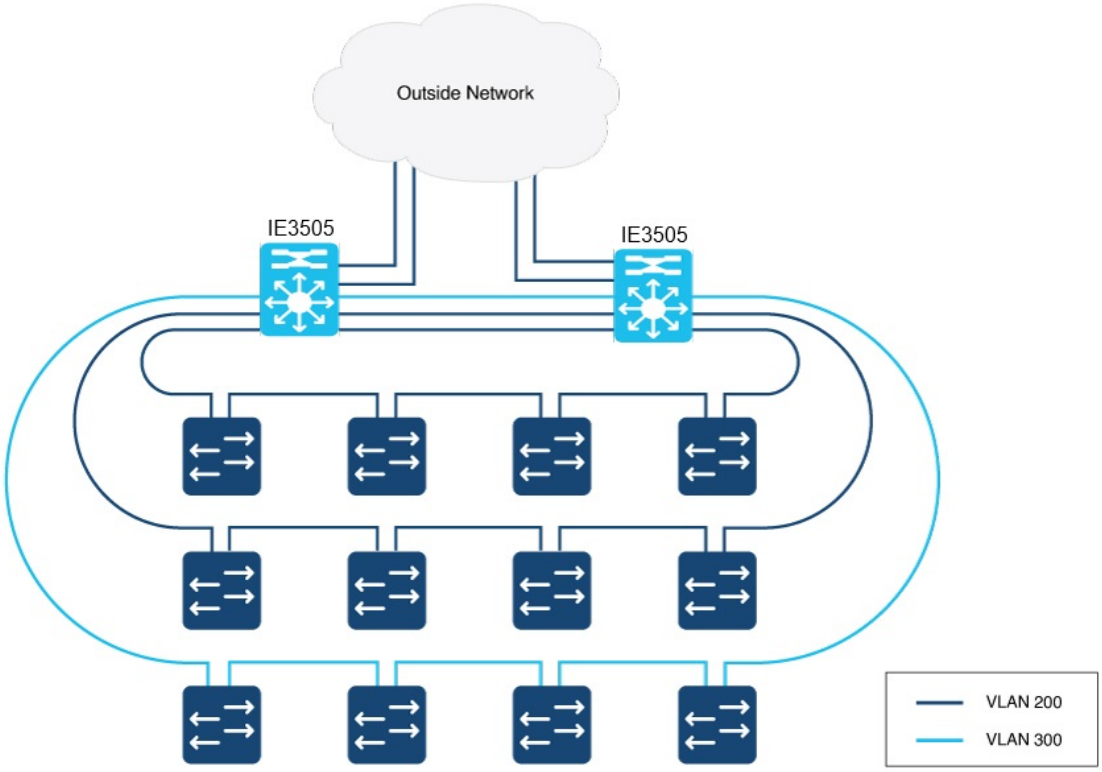
Multiple Rings Connected to Multiple Switches

You can also use multiple rings with multiple Cisco IE3500 Series Switches, as shown in the illustration below. Depending on the configuration of the switches, VLAN restrictions can apply.

If the two switches are configured as redundant gateways for the same set of rings, there are no VLAN restrictions. The following example shows two rings on the same VLAN and one ring on a separate VLAN. However, because there are no VLAN restrictions, you can also configure all three rings on the same VLAN or all three on separate VLANs.

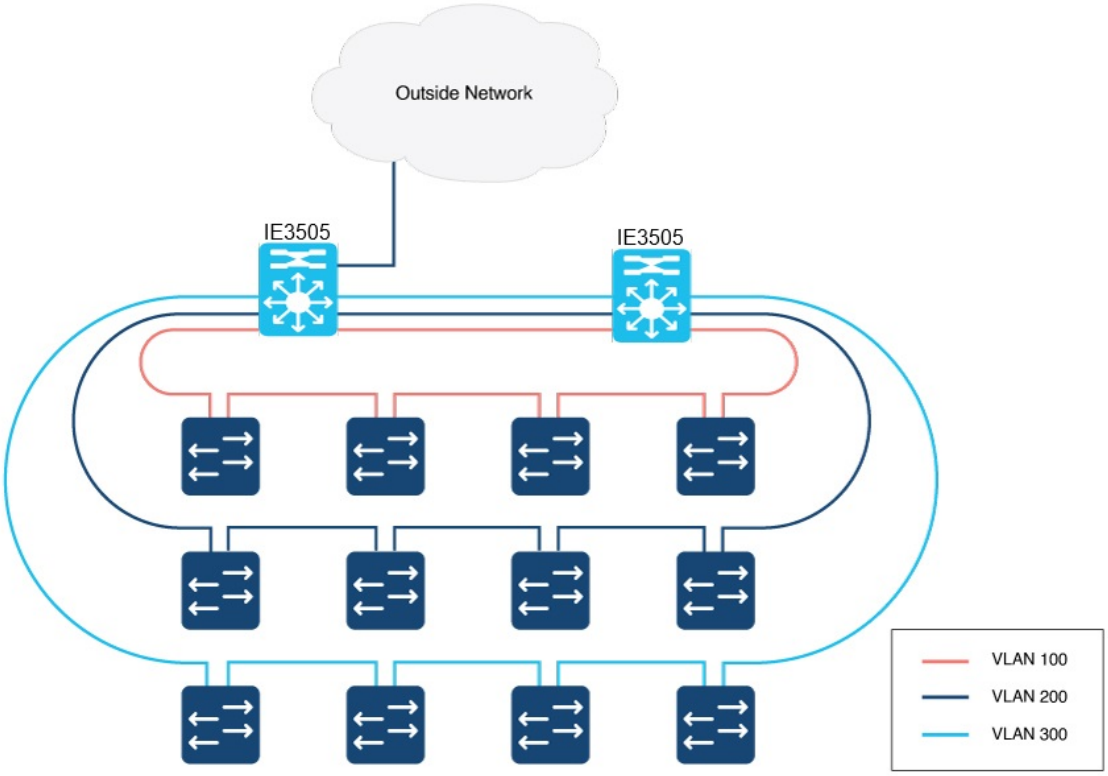
When there are two or more switches in same DLR, and they are not configured as redundant gateways, then each DLR ring must have a unique VLAN. VLANs and mutiple DLR rings need to be planned, especially when the VLAN or VLANs are present on more than one DLR ring. Configuring redundant gateways on the IE switches enables DLR deployments where a VLAN is present on multiple DLR rings. When the DLR gateway is not configured on the IE switch pair, then a VLAN cannot be shared across rings. Failure to adhere to this guidance will result in a Layer 2 loop. In example below, the two IE switches are configured for DLR gateway, thus a VLAN can be present across two or more DLR rings.

Figure 191: Multiple Rings, Multiple Switches, No VLAN Restrictions



If the two switches are not configured as redundant gateways, VLANs cannot be present on more than one ring, otherwise a Layer 2 loop becomes possible. The following illustration shows only one path out of the DLR ring, so DLR redundant gateways have not been configured.

Figure 192: Vlans Not Shared Across DLR Rings



Redundant Gateways

A DLR network with redundant gateways uses multiple switches to provide multiple connections from a ring to the outside network. Redundant gateways are not essential if you need only one connection to the outside network, but they provide extra network resiliency if an uplink connection fails.

Either a ring supervisor or a ring participant can be a redundant gateway; however, you must enable and configure DLR on both gateway switches.

Redundant gateways enable you to automatically or manually chose an active gateway as well as for automatic switchover to a backup gateway in case of a connection failure. Gateway switchover times range from 14 ms to 6.1 seconds, depending on the uplink network resiliency protocol. DLR redundancy gateway performance applies to traffic sourced from inside the DLR destined to the outside network:

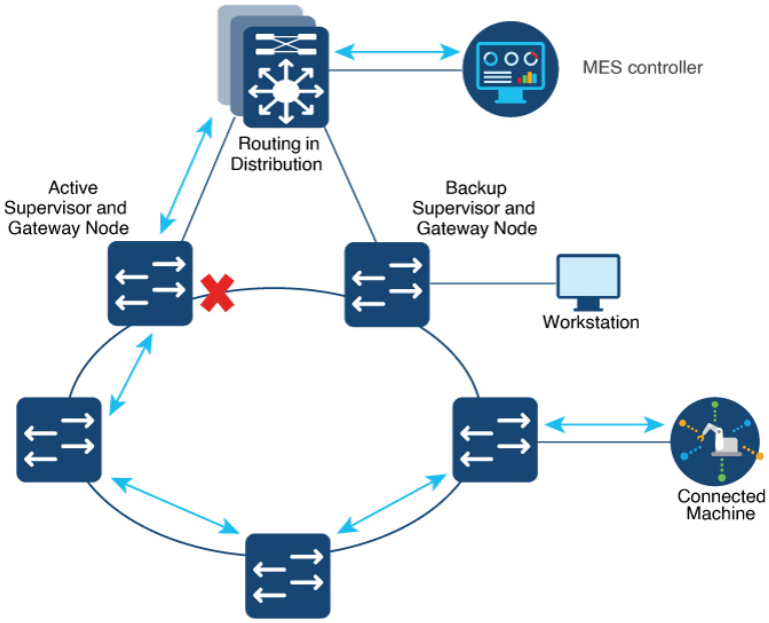
- Uplink connection failure detected by the active gateway at the physical layer is anywhere from 14 to 150 ms.
- Failure of the Active Gateway Node can take between 19 and 150 ms.

System performance, which applies to most applications, describes traffic sourced from the outside network destined to the DLR. Higher layer uplink fault detection is up to 6.1 seconds.

DLR gateway convergence depends on the redundancy protocol running on the gateway interfaces. STP and REP have different convergence times. Traffic in and out of the DLR ring to the outside network should converge on failure to match the protocol used.

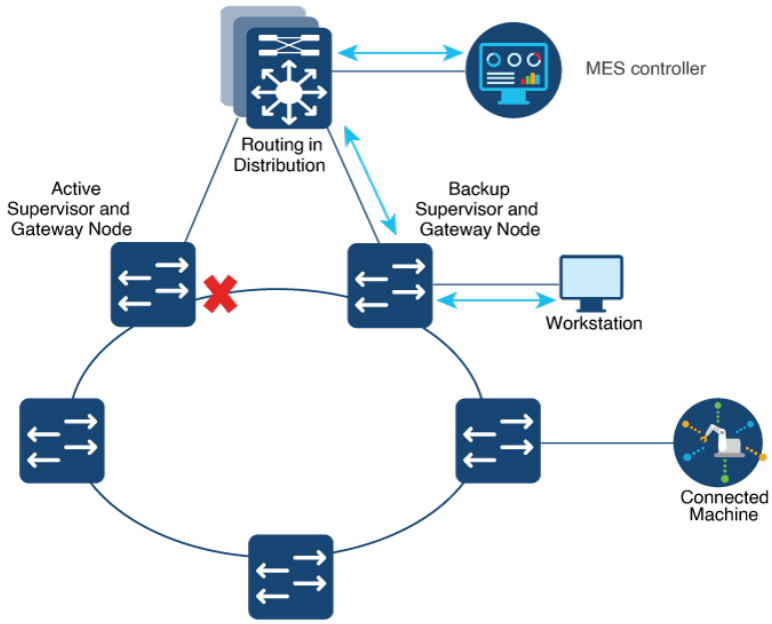
The following illustration shows traffic flow in and out of a DLR ring through the active DLR gateway.

Figure 193: DLR Active Gateway Traffic Flow



The following illustration shows traffic flow in and out of a DLR ring through the backup DLR gateway for devices directly connected to the backup gateway node. It is important to understand the physical path devices take to communicate with other applications outside the ring. The devices attached to the configured backup gateway take a different path than devices attached to the active gateway or other nodes in the DLR ring.

Figure 194: DLR Backup Gateway Traffic Flow



For more information about redundant gateways, see *Guidelines for Using Device Level Ring (DLR) with Ethernet/IP*™ on the odva.org website.

Cisco IE Switch Support for DLR

The Cisco IE3505 Rugged Series Switches and IE3505 Heavy-Duty Series Switches support DLR.

Switches that Support DLR

Table 194: The following Advanced base modules SKUs (PIDs) supports DLR.

Switch	PID
Cisco IE3505 Rugged Series Switch	IE-3505-8P3S
	IE-3505-8T3S
Cisco IE3505 Heavy-Duty Series Switch	IE-3505H-16T

Table 195: The following Expansion modules PIDs include DLR support when installed on one of the above advanced base modules

Switch	Expansion modules PID
Cisco IE3500 Series Switch	IEM-3500-14T2S
	IEM-3500-6T2S
	IEM-3500-16P
	IEM-3500-16T
	IEM-3500-8P
	IEM-3500-8S
	IEM-3500-8T

Support for DLR is available on Network Essentials and Network Advantage licenses.

Supported DLR Features

IE-3505-8P3S, IE-3505-8T3S, and IE-3505H-16T switches support the following DLR features:

- Three DLR rings for each switch and expansion model combination as shown in the following table.

  Table 196: Ring Support for Cisco IE3505 Series Switch and Expansion Models

Switch	FPGA Profile	Number of Rings
Cisco IE3505 Rugged Series without expansion module	Default	3
	Redundancy	2
Cisco IE3505 Rugged Series with expansion module	Default	3
	Redundancy	2
IE-3505H-16T	Default	3
	Redundancy	2

- Redundant gateway
- Web User Interface (WebUI): DLR can be configured using the WebUI.
- Common Industrial Protocol (CIP): DLR can be configured using CIP.

Differences Between Switch Models When Using DLR

Port mapping for IE3505 Heavy-Duty Series Switches differs from port mapping for IE3505-8P3S and IE3505-8T3S Rugged Series Switches.

For IE3505-8P3S and IE3505-8T3S Rugged Series Switches, DLR is supported on the base system (Gi1/1, Gi1/2), (Gi1/4, Gi1/5), (Gi1/6, Gi1/7), (Gi1/8, Gi1/9), and (Gi1/10, Gi1/11), and on the expansion module Gi2/1 through Gi2/8.

Table 197: Cisco IE3505 Rugged Series Switch Port Pairing

Module	Valid DLR Ring Port Pairs
Base module	• [Gi1/1, Gi1/2] • [Gi1/4, Gi1/5] • [Gi1/6, Gi1/7] • [Gi1/8, Gi1/9] • [Gi1/10, Gi1/11]
Expansion module	• [Gi2/1, Gi2/2] • [Gi2/3, Gi2/4] • [Gi2/5, Gi2/6] • [Gi2/7, Gi2/8]

For Cisco IE3505 Heavy-Duty Series Switches, you must pair DLR ring ports with adjacent ports. The following table shows valid DLR ring port pairs:

Table 198: Cisco IE3505 Heavy-Duty Switch Port Pairing

Ports	Valid DLR Ring Port Pairs
Gi1/1 through Gi1/8	• [Gi1/1, Gi1/2] • [Gi1/3, Gi1/4] • [Gi1/5, Gi1/6] • [Gi1/7, Gi1/8]
Gi1/9 through Gi1/16	• [Gi1/9, Gi1/10] • [Gi1/11, Gi1/12] • [Gi1/13, Gi1/14] • [Gi1/15, Gi1/16]

DLR Feature Interactions

The following list contains features that cannot be configured on interfaces that are also configured as DLR ring ports..

- STP, RSTP, and MSTP
- 802.1x and Guest VLAN
- PVLAN and PVLAN Edge

- VLAN Routing and Bridging and MV

DLR does not interfere with the functionality of the following features. However, take care during configuration: The MAC or IP addresses of the DLR devices must be included in the allowable list.

- Port Security
- Unicast MAC filter
- DAI
- DHCP Snooping

For the following features, the ports forward IGMP packets between the two DLR ports but do not process them. Devices other than gateways and active redundant gateway devices are unaffected.

- Multicast
- IGMP Snooping

Guidelines and Limitations

The following restrictions apply to DLR configuration and operation:

- You can configure up to three DLR rings at the same time on each the Cisco IE3505 Rugged Series Switches and IE3505 Heavy-Duty Series Switches. See use cases in Multiple Rings, on page 2641 for guidance.
- When configuring DLR Gateways, for each node, you can configure two ports as an uplink. An uplink can belong to more than one ring.
- We recommend that you configure no more than one backup gateway for each ring.
- MAC learning for each ring is limited to 1024 unicast MAC addresses and 128 multicast MAC addresses for each he Cisco IE3505 Rugged Series Switches and IE3505 Heavy-Duty Series Switches.
- Multicast MAC learning through IGMP snooping is limited to 128 addresses.
- Duplicated packets may be observed during ring convergence.
- DLR is supported on 1 Gbps links and 100 Mbps interfaces with full duplex capability. DLR does not support half duplex links.
- On a given physical ring, all the nodes must be configured with same ring- ID If there is any mismatch in ring IDs between nodes (due to misconfiguration), the ring will still converge and may lead to unexpected behavior.

The following restrictions apply to configuring multiple DLR rings:

- Multiple rings cannot share the same ring ports.
- The switch cannot be configured as announce-based node.
- The default beacon interval is 400 usec. This is the recommended interval for 1 Gbps and 100 Mbps ring interface speeds. The default beacon timeout is 1960 usec. This is the recommended value.

- DLR ring ports are supported on IEM-3500 expansion modules. Check the Product ID (PID) of any expansion modules attached to the IE3505 before attempting to configure DLR rings on expansion module ports.

Note For information, including limitations, on DLR interactions with other features and protocols, see the section DLR Feature Interactions, on page 2649 in this guide.

Configuring DLR

The following sections provide information for configuring DLR on IE3505 Rugged Series Switches and IE3505 Heavy-Duty Series Switches. The supervisor node with the highest precedence value is elected to operate as DLR supervisor. You can use this feature to plan which node will be active and which will be backup supervisor.

Configure a Ring Supervisor

Complete the following procedure to configure the switch as a ring supervisor.

Before you begin

Refer to the parameters for configuring a DLR ring supervisor, which are shown in the following table.

Parameter	Range	Default
Beacon interval	200 to100,000 microseconds	400 microseconds
Beacon timeout	200 to 500,000 microseconds	1960 microseconds
Precedence	0 to 255	0
control-vlan-id	0 to 4095	0

Procedure

	Command or Action	Purpose
Step 1	**dlr ring** *ring_number* **Example:** `switch(config)#dlr ring 1`	Provide the unique DLR value identifying a ring.
Step 2	**mode** *device_role* **Example:** `switch(config)#mode supervisor`	Configure the DLR device as a ring supervisor.
Step 3	**beacon-interval** *microseconds* **Example:**	Set the beacon interval. **Note**

	Command or Action	Purpose
	`switch(config)#beacon-interval 500`	You can set the beacon interval only for devices in supervisor mode.
Step 4	**beacon-timeout** *microseconds* **Example:** `switch(config)# beacon-timeout 2500`	Set the beacon timeout. **Note** You can set the beacon timeout only for devices in supervisor mode.
Step 5	**precedence** *rank* **Example:** `switch(config)#precedence 100`	Sets the precedence of the ring supervisor.
Step 6	**interface** *interface_name* **Example:** `switch(config)#interface gigabitEthernet 1/1`	Enter interface configuation submode for interface GigabitEthernet 1/1.
Step 7	**switchport mode access** **Example:** `switch(config-if)#switchport mode access` `switch(config-if)#switchport access vlan 33`	Configure the interface to be a member of a single VLAN.
Step 8	**dlr ring 1** **Example:** `switch(config-if)#dlr ring 1`	Configure the interface to be a member of a DLR ring.
Step 9	**interface** **Example:** `switch(config)#interface gigabitEthernet 1/2`	Set the interface for the second DLR ring port. The second DLR ring port must be a valid port pair of the first DLR ring port. See the section Cisco IE Switch Support for DLR, on page 2647 in this guide for valid port pair combinations.
Step 10	**switchport mode access** **Example:** `switch(config-if)#switchport mode access` `switch(config-if)#switchport access vlan 33`	Configure the interface to be a member of a single VLAN. The VLAN must be the same as the one used by the other interface to be a port on same DLR ring.
Step 11	**dlr ring 1** **Example:** `switch(config-if)#dlr ring 1`	Add interface for the DLR ring port.

What to do next

Verify that the ring supervisor is configured by entering the show command. The following example is output of the show command when the switch is configured as a ring supervisor:

```
Switch#sh dlr ring 1
DLR ring 1

mode: Active Supervisor
Network Topology:     Ring Network Status: Normal
IOS state: NORMAL_ACTIVE    Hardware State: NORMAL_ACTIVE
Transition bit = 0
Mac-Addr: 6C:13:D5:AC:3A:C3 IP-Addr: 0.0.0.0
Port1: GigabitEthernet1/1, vlan 33,   UP Port2: GigabitEthernet1/2, vlan 33, UP
LastBcnRcvPort: Port 1: Yes    Port 2: Yes

Active Supervisor Parameters:
Beacon Interval (usec): 500    Beacon Timeout (usec): 2500
DLR VLAN ID: 0      Precedence: 100
Mac-Addr: 6C:13:D5:AC:3A:C3    IP-Addr: 0.0.0.0

Locally Configured Supervisor Parameters:
Beacon Interval (usec): 500    Beacon Timeout (usec): 2500
DLR VLAN ID: 0      Precedence: 100
Port1: GigabitEthernet1/1    Port2: GigabitEthernet1/2
```

Configure a Beacon-Based Ring Node

Procedure

Complete the commands as shown in the following example to configure the switch as a beacon-based ring node.

Example:

```
…
dlr ring 2
   mode beacon-node
!
…
interface GigabitEthernet1/1
   switchport mode trunk
   dlr ring 2
!
interface GigabitEthernet1/2
   switchport mode trunk
   dlr ring 2
!
…
```

What to do next

Verify that the beacon-based ring node is configured by entering the show command. The following example is output of the show command when the switch is configured as a beacon-based ring node:

```
Switch#show dlr ring 2
DLR ring 2
mode: Beacon Node
Network Topology: Ring     Network Status: Normal
IOS state: NORMAL     Hardware State: NORMAL
Transition bit = 0
Mac-Addr: 6C:13:D5:AC:3C:03 IP-Addr: 0.0.0.0
```

```
Port1: GigabitEthernet1/1, vlan Trunk, UP  Port2: GigabitEthernet1/2, vlan Trunk, UP
LastBcnRcvPort: Port 1: Yes    Port 2: Yes

Active Supervisor Parameters:
Beacon Interval (usec): 400    Beacon Timeout (usec): 1960
DLR VLAN ID: 0 Precedence: 0
Mac-Addr: 6C:13:D5:AC:3A:C3    IP-Addr: 0.0.0.0

Locally Configured Beacon Node Parameters:
Port1: GigabitEthernet1/1    Port2: GigabitEthernet1/2
```

Configure a Redundant Gateway

You must configure DLR on both gateway switches.

Before you begin

Refer to the parameters for configuring a switch as a DLR redundant gateway node. The parameters are shown in the following table:

Parameter	Range	Default
Gateway enable	Enable-Disable	Disable
Precedence	0 to 255	0
Advertise interval	200 to 100,000 microseconds	2000 microseconds
Advertise timeout	500 to 500,000 microseconds	5000 microseconds
Learning-update	Supported	Enabled

Procedure

Complete the commands as shown in the following example to configure the switch as a redundant gateway node.

Example:

Switch A Configuration	Switch B Configuration
… dlr ring 1 mode supervisor dlr ring 1 gateway enable gateway-precedence 100 advertise-interval 3000 advertise-timeout 10000 interface GigabitEthernet1/9 switchport mode trunk dlr ring 1 uplink !…	… dlr ring 1 mode supervisor dlr ring 1 gateway enable gateway-precedence 255 advertise-interval 3000 advertise-timeout 10000 interface GigabitEthernet1/9 switch mode trunk dlr ring 1 uplink !…

What to do next

Verify that the redundant gateways are configured by entering the show command.

The following example is output of the show command when a switch is configured as the redundant gateway nodes:

```
Switch-a#sh dlr ring 1
DLR ring 1

mode: Active Supervisor
Network Topology: Ring    Network Status: Normal
IOS state: NORMAL_ACTIVE   Hardware State: NORMAL_ACTIVE
Transition bit = 0
Mac-Addr: 6C:13:D5:AC:3C:03 IP-Addr: 0.0.0.0
Port1: GigabitEthernet1/1, vlan Trunk, UP Port2: GigabitEthernet1/2, vlan Trunk, UP
LastBcnRcvPort: Port 1: Yes   Port 2: Yes

Active Supervisor Parameters:
Beacon Interval (usec): 400   Beacon Timeout (usec): 1960
DLR VLAN ID: 0     Precedence: 0
Mac-Addr: 6C:13:D5:AC:3C:03 IP-Addr: 0.0.0.0

Locally Configured Supervisor Parameters:
Beacon Interval (usec): 400   Beacon Timeout (usec): 1960
DLR VLAN ID: 0     Precedence: 0
Port1: GigabitEthernet1/1   Port2: GigabitEthernet1/2
…
…
…
Redundant Gateway Information:
Redundant Gateway Status: Active Gateway
Hardware State: ACTIVE NORMAL
Mac-Addr: 6C:13:D5:AC:3C:03 IP_addr:0.0.0.0
Uplink Port(s): GigabitEthernet1/9

Active Gateway Parameters:
Advertise Interval (usec): 3000  Advertise Timeout (usec): 10000
Precedence: 100    Learning Update Enable: yes
Mac-Addr: 6C:13:D5:AC:3C:03 IP-Addr:0.0.0.0

Fault Statistics:
Gateway Faults since power up: 0
```

```
Locally Configured Gateway Parameters:
Advertise Interval (usec): 3000  Advertise Timeout (usec): 10000
Precedence: 100    Learning Update Enable: yes
Uplink Port(s): GigabitEthernet1/9
switch-a#
```

The following example is output of the show command when a switch is configured as the backup gateway:

```
Switch-b#sh dlr ring 1
----------------------------------------
DLR ring 1

mode: Backup Supervisor
Network Topology: Ring    Network Status: Normal
IOS state: NORMAL_BACKUP   Hardware State: NORMAL_BACKUP
Transition bit = 0
Mac-Addr: 6C:13:D5:AC:3A:C3 IP-Addr: 0.0.0.0
Port1: GigabitEthernet1/1, vlan Trunk, UP Port2: GigabitEthernet1/2, vlan Trunk, UP
LastBcnRcvPort: Port 1:   Yes Port 2: Yes

Active Supervisor Parameters:
Beacon Interval (usec): 400   Beacon Timeout (usec): 1960
DLR VLAN ID: 0     Precedence: 0
Mac-Addr: 6C:13:D5:AC:3C:03 IP-Addr: 0.0.0.0

Locally Configured Supervisor Parameters:
Beacon Interval (usec): 400   Beacon Timeout (usec): 1960
DLR VLAN ID: 0     Precedence: 0
Port1: GigabitEthernet1/1   Port2: GigabitEthernet1/2
…
…
…
Backup Supervisor Precedence: 0

Redundant Gateway Information:
Redundant Gateway Status: Backup Gateway
Hardware State: BACKUP NORMAL
Mac-Addr: 6C:13:D5:AC:3A:C3 IP_addr:0.0.0.0
Uplink Port(s): GigabitEthernet1/9

Active Gateway Parameters:
Advertise Interval (usec): 3000  Advertise Timeout (usec): 10000
Precedence: 100    Learning Update Enable: yes
Mac-Addr: 6C:13:D5:AC:3C:03 IP-Addr:0.0.0.0

Fault Statistics:
Gateway Faults since power up: 0

Locally Configured Gateway Parameters:
Advertise Interval (usec): 3000  Advertise Timeout (usec): 10000
Precedence: 0     Learning Update Enable: yes
Uplink Port(s): GigabitEthernet1/9
```

Configure VLAN Trunking

Note When a node has two or more DLR rings configured, a VLAN can only be present on one ring. When configuring DLR ring ports in trunk mode, you must edit the trunk-allowed VLAN list to ensure unique VLAN membership across DLR rings.

Procedure

Complete the commands as shown in the following example to configure VLAN trunking for DLR.

Example:

```
switch(config)#dlr ring 1
switch(config-dlr)#mode supervisor
switch(config-dlr-supervisor)#end

switch(config)#int gi1/1
switch(config-if)#switchport mode trunk
switch(config-if)#switchport trunk allowed vlan 10,20
switch(config-if)#dlr ring 1

switch(config-if)#
switch(config-if)#int gi1/2
switch(config-if)#switchport mode trunk
switch(config-if)#switchport trunk allowed vlan 10,20
switch(config-if)#dlr ring 1
```

What to do next

Verify that VLAN trunking is configured by entering the show command. The following example is the output of the show command when VLAN trunking is configured:

```
switch#sh dlr ring
----------------------------------------
DLR ring 1

mode: Active Supervisor
Network Topology: Ring     Network Status: Normal
IOS state: NORMAL_ACTIVE    Hardware State: NORMAL_ACTIVE
Transition bit = 0
Mac-Addr: 6C:13:D5:AC:3A:C3 IP-Addr: 0.0.0.0
Port1: GigabitEthernet1/1, vlan Trunk, UP  Port2: GigabitEthernet1/2, vlan Trunk, UP
LastBcnRcvPort: Port 1: Yes    Port 2: Yes

Active Supervisor Parameters:
Beacon Interval (usec): 400    Beacon Timeout (usec): 1960
DLR VLAN ID: 0      Precedence: 0
Mac-Addr: 6C:13:D5:AC:3A:C3 IP-Addr: 0.0.0.0

Locally Configured Supervisor Parameters:
Beacon Interval (usec): 400    Beacon Timeout (usec): 1960
DLR VLAN ID: 0      Precedence: 0
Port1: GigabitEthernet1/1    Port2: GigabitEthernet1/2

Ring Protocol Participants Count: 3
No    Mac-Addr IP-Addr
1     6C:13:D5:AC:3A:C3 0.0.0.0
2     6C:13:D5:AC:3C:03 0.0.0.0
3     6C:13:D5:AC:37:03 0.0.0.0

Fault Statistics:CIP

Ring Faults since power up: 0
Ring Fault Location  Mac-Addr IP-Addr
```

```
Last Active Node on Port 1 00:00:00:00:00:000.0.0.0
Last Active Node on Port 2 00:00:00:00:00:000.0.0.0

Redundant Gateway Information:
Redundant Gateway Status: Gateway not enabled
---------------------------------------
DLR ring 2 not configured
```

Enabling CIP

You can enable Common Industrial Protocol (CIP) on a device by applying the `cip enable` command on one of the Layer-3 interfaces—a physical L3 interface or an SVI-interface.

Note Be aware of the following when enabling CIP:

- You must habve DLR rings configured on the switch before enabling CIP.
- You must enter the command in interface configuration mode.
- You enable CIP at the device level.
- You enable CIP only through one of the Layer-3 interfaces; if you try to enable CIP on another interface, an error occurs.

Enable CIP on the Layer 3 Interface

Complete the steps in this section to enable CIP on the Layer 3 interface.

Procedure

	Command or Action	Purpose
Step 1	**conf t**	Enter configuration mode.
Step 2	**interface** *interface_name* **Example:** `switch(config)#interface gigabitEthernet 1/10`	Specify the interface.
Step 3	**no switchport**	Prevent the interface from forwarding Ethernet frames based on MAC addresses. The interface is not operational until a valid IP address is assigned.
Step 4	**ip address** *IP_address subnet_address* **Example:** `switch(config-if)#ip address 192.168.1.10 255.255.255.0`	Set the IP address and subnet.

	Command or Action	Purpose
Step 5	**cip enable**	Enable CIP on the interface.
Step 6	**end**	Leave configuration mode.

What to do next

Verify that CIP is configured by entering the show command. The following example is output of the `show` command when CIP is configured:

```
DLR_node#show cip status
State : Enabled
Interface : Gi1/10
DLR_node#
```

Enable CIP on the SVI Interface

Complete the steps in this section to enable CIP on the SVI interface.

Before you begin

If the SVI is not vlan1, assign `switchport access vlan` *`vlan-id`* to the DLR ring.

Procedure

	Command or Action	Purpose
Step 1	**conf t**	Enter configuration mode.
Step 2	**vlan** *vlan_id* **Example:** `switch(config)#vlan 1`	Specify the VLAN.
Step 3	int vlan *vlan_id* **Example:** `switch(config-vlan)#int vlan 1`	Enter interface configuration mode.
Step 4	**ip address** *IP_address subnet_address* **Example:** `switch(config-if)#ip address 192.168.1.10 255.255.255.0`	Specify an ID address and subnet.
Step 5	**cip enable** **Example:** `switch(config-if)# cip enable`	Enable CIP on the interface.
Step 6	**end** **Example:** `switch(config-if)# end`	Leave configuration mode.

What to do next

Verify that CIP is configured by entering the show command. The following example is output of the `show` command when CIP is configured:

```
DLR_node#show cip status
State : Enabled
Interface : Vlan 1
DLR_node#
```

PART XI

Common Industrial Protocol

CHAPTER 177

Common Industrial Protocol

Information About CIP

The Common Industrial Protocol (CIP) is an industrial protocol for industrial automation applications.Previously known as Control and Information Protocol, CIP encompasses a comprehensive suite of messages and services for the collection of manufacturing automation applications - control, safety, synchronization,motion, configuration and information

It is supported by Open DeviceNet Vendors Association (ODVA), an organization that supports network technologies based upon CIP such as DeviceNet, EtherNet/IP, CIP Safety and CIP Sync. CIP allows users to integrate these manufacturing applications with enterprise-level Ethernet networks and the Internet.

CIP Restrictions

CIP can be enabled on only one VLAN on the switch.

Enabling CIP

Before you begin

By default, CIP is not enabled

Procedure

Step 1 Enters global configuration mode.

```
Configure Terminal
```

Step 2 Sets CIP security options on the switch.

```
cip security { password password | window timeout value }
```

Step 3 Enters interface configuration mode.

```
interface vlan 20
```

Step 4 Enables CIP on a VLAN.

```
cip enable
```

Step 5 Returns to privileged EXEC mode.

```
end
```

Step 6 Verifies your entries.

```
show running-config
```

Step 7 (Optional) Saves your entries in the configuration file.

```
copy running-config startup-config
```

Step 8 (Optional) Displays information about the CIP subsystem.

```
show cip { connection | faults | file | miscellaneous |
object | security | session | status }
```

Step 9 (Optional) Enables debugging of the CIP subsystem.

```
debug cip {assembly | connection manager | dlr | errors
| event | file | io | packet | infra | security | session | socket}
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOSbasic commands	Cisco IOS Configuration Fundamentals Command Reference

Standards and RFCs

Standard/RFC
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.

Technical Assistance

Standard/RFC	Title
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	https://www.cisco.com/support

PART XII

PROFINET

CHAPTER 178

PROFINET

Information About Configuring PROFINET

PROFINET is the leading Industrial Ethernet standard that uses TCP/IP and IT standards to connect and control machines in real time. It is widely used in industrial automation and process control networks, especially for motion control and precision instrumentation. PROFINET emphasizes fast, reliable data exchange and defines communication paths to meet different speed requirements.

Conformance Classes: PROFINET has different conformance classes that define supported features.

- **Class B:** Common in factory automation, it supports fast, real-time communication and diagnostics — ideal for applications like production lines and equipment monitoring. Class B includes **PROFINET Real Time (RT)**, which prioritizes important data to reduce delays, with cycle times around 10 ms. This makes RT suitable for tasks like conveyor belt and packaging machine control. However, the switches do not support **Isochronous Real Time (IRT)**, which is required for ultra-precise synchronization.

Communication Levels: PROFINET communication is scalable across three levels:

- **Non-Real-Time (NRT):** Uses TCP/IP with bus cycle times around 100 ms.
- **Real-Time (RT):** Enables faster cycle times, approximately 10 ms.
- **Isochronous Real-Time (IRT):** Achieves highly precise synchronization with cycle times as low as 1 ms (not supported in Class B).

PROFINET I/O System: PROFINET I/O is a flexible communication framework for distributed automation. It uses cyclic data transfer to exchange information, alarms, and diagnostics between controllers, I/O devices, and automation systems like motion controllers.

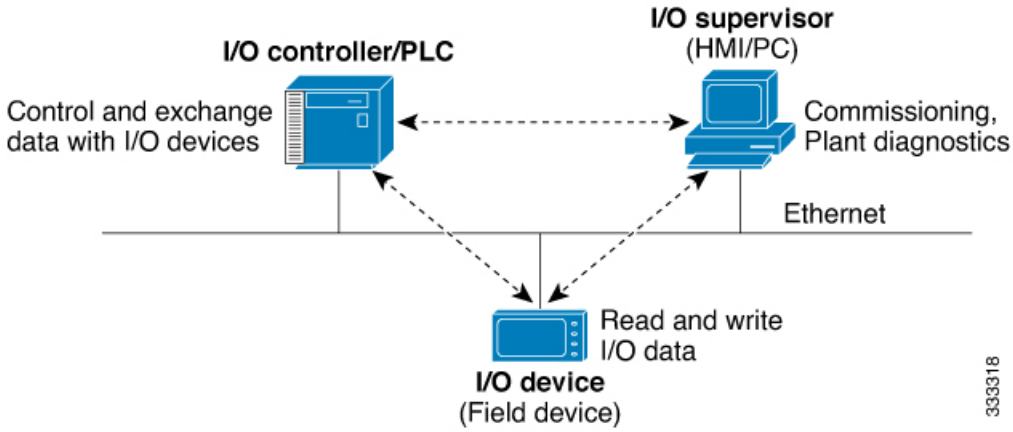
PROFINET Device Roles

An I/O controller is a programmable logic controller (PLC) that controls I/O devices and exchanges data such as configuration, alarms, and I/O data through an automation program. The I/O controller and the I/O supervisor exchange diagnostic information. The I/O controller shares configuration and I/O information with an I/O device and receives alarms from the I/O device.

There are three main types of I/O devices:

- **I/O Devices:** Field devices connected to controllers.
- **I/O Controllers:** Manage and control field devices.
- **I/O Supervisors:** Tools for diagnostics and configuration.

Figure 195: PROFINET Device Roles



PROFINET is designed to serve as the sole or primary management system platform for industrial networks. It streamlines device configuration and communication, reducing the need for manual setup.

Automatic Device Detection and Configuration: The I/O controller automatically detects switches using the Discovery and Configuration Protocol (DCP). It sets the device name and IP address, so you don't need to enter Cisco IOS commands for basic configurations.

Advanced Configuration with Cisco IOS: For advanced features, like Quality of Service (QoS), DHCP, and similar capabilities, you'll need to configure the switch using Cisco IOS commands. These advanced settings cannot be managed directly through PROFINET.

An I/O supervisor is an engineering station, such as a human machine interface (HMI) or PC, used for commissioning, monitoring, and diagnostic analysis. The I/O supervisor exchanges diagnostic, status, control, and parameter information with the I/O device.

An I/O device is a distributed I/O device such as a sensor, an actuator, or a motion controller.

Note If Profinet DCP cannot detect the switch, PLC, or IO mac addresses, temporarily disable the firewall or virus scan from the Windows PC that installed the Siemens STEP7 or TIA Portal Automation application.

In a PROFINET I/O system, all the I/O devices communicate over an Ethernet communication network to meet the automation industry requirement for bus cycle times of less than 100 ms. The network uses switches and full-duplex data exchange to avoid data collisions. The switches play the role of I/O device in Figure 195: PROFINET Device Roles, on page 2670.

PROFINET Device Data Exchange

After PROFINET uses DCP to discover devices, including the switch, the devices establish application relationships (ARs) and communication relationships (CRs). After a connection is established and information about device parameters is exchanged, I/O data is exchanged. The switch uses non-real-time CRs to exchange the data attributes listed below.

Table 199: PROFINET I/O Switch Attributes

PROFINET I/O Switch Configuration Attributes	Value or Action
Device name	Configures a name for the device.
TCP/IP	IP address, subnet mask, default gateway, and switch virtual interface (SVI).
Primary temperature alarm	Enables or disables monitoring for the specified alarm.
Secondary temperature alarm	Enables or disables monitoring for the specified alarm.
SD card alarm	Enables or disables monitoring for the specified alarm.
RPS failed alarm	Enables or disables monitoring for the redundant power supply alarm. **Note** Before enabling monitoring for the RPS failed alarm, the user must configure the command "power-supply dual" in CLI to trigger an alarm when one of the power supplies is missing or inoperable.
Relay major alarm	Enables or disables monitoring for the specified alarm.
Reset to factory	Reset to factory (Mode 2: Communication parameter) Uses the PROFINET I/O controller to reset the switch communication parameter. This action removes the Profinet device ID, IP address, and configured SNMP writable MIBs.
Relay major configuration	Specifies the type of port alarm, for example, link fault, that triggers the major relay. Any port configured with the specified alarm type can trigger the major relay.

Table 200: PROFINET I/O Port Attributes

PROFINET I/O Port Configuration Attributes	Value or Action
Speed	10, 100, 1000, or auto.
Duplex	half, full, or auto.
Port mode	access or trunk.
Link status	shut down or no shut down.
Configure rate limiting	Broadcast, unicast, or multicast threshold exceeds configured levels.
Port link fault alarm	Enables or disables monitoring for specified alarm.
Port not forwarding alarm	Enables or disables monitoring for specified alarm.
Port not operating alarm	Enables or disables monitoring for specified alarm.
Port FCS threshold alarm	Enables or disables monitoring for specified alarm.

General Station Description File

PROFINET devices are integrated by using a general station description (GSD) file that contains the data for engineering and data exchange between the I/O controller, the I/O supervisor, and the I/O devices, including the switch. Each PROFINET I/O field device must have an associated GSD file that describes the properties of the device and contains all this information required for configuration:

- Device identification information (device ID, vendor ID and name, product family, number of ports)
- Number and types of pluggable modules
- Error text for diagnostic information
- Communication parameters for I/O devices, including the minimum cycle time, the reduction ratio, and the watch dog time
- Configuration data for the I/O device modules, including speed, duplex, VLAN, port security information, alarms, and broadcast rate limiting thresholds
- Parameters configured for I/O device modules for the attributes listed above

The PROFINET GSD file is bundled with the Cisco IOS release. After the switch boots at least one time, the GSD files for the switch are located in a directory called ProfinetGSD. In this directory, there is a zip file containing all the GSDs for all Cisco IE3500 Series Switch SKUs. The file is called CISCO_*product_id*.zip, for example, CISCO_IE35xx.zip.

The GSD file is in the switch and the I/O supervisor uses this file to manage the switch. For IOS XE-based platforms, the GSD file can be found in the Flash: or SDFlash: file system. If you want to load the GSD file for the Cisco IOS XE platform into the I/O supervisor, you need to copy it from the switch.

Note You must use the GSD file that is associated with the Cisco IOS release on the switch to manage your PROFINET network. Both the I/O supervisor and the Cisco IOS software alert you to a mismatch between the GSD file and the switch's Cisco IOS software version.

The status of GSD match or mismatch can be determined using the **show profinet status** command.

Configuring PROFINET

You can use either the SIMATIC STEP7 or TIA Portal Automation application on the I/O supervisor, or you can use the Cisco IOS software to configure PROFINET on the switch.

After you enable PROFINET, Link Layer Discovery Protocol (LLDP) is automatically enabled on the switch because PROFINET relies on LLDP to fully function. If you disable PROFINET, you can enable or disable LLDP as needed.

Configure the default PROFINET settings on a switch

This task explains how to activate PROFINET on a switch by enabling the default VLAN 1 configuration.

PROFINET is enabled by default on all switches. The default configuration operates on VLAN 1, but you can assign it to another VLAN ID if required. By default, VLAN 1 is in a shutdown state when the switch is first powered on. To activate PROFINET on an out-of-the-box switch, you must unshut VLAN 1.

VLAN 0 Priority Tagging

Starting with Cisco IOS XE Release 17.18.1, the VLAN 0 Priority Tagging feature allows you to prioritize traffic without assigning it to a specific VLAN. This feature enables the transmission and reception of 802.1Q Ethernet frames with the VLAN ID set to zero, by retaining the 802.1P priority bits of the VLAN 0 Ethernet packets. These Ethernet frames are known as **priority tagged** frames. As a result, critical traffic receives higher processing priority.

Procedure

Step 1 Enter the global configuration mode, with the **configure terminal** command.

Example:

```
Switch#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
```

Step 2 Access the VLAN 1 interface with the **interface vlan 1** command.

Example:

```
Switch(config)#interface vlan 1
```

Specify VLAN 1 as the interface to be configured

Step 3 Enable VLAN 1 with the **no shut** command.

Example:

```
Switch(config-if)#no shut
```

Step 4 Exit the configuration mode.

Example:

```
Switch(config-if)#end
```

If PROFINET has been disabled, follow the specific procedure for enabling PROFINET on the switch in Enabling PROFINET.

Note
Cisco devices undergo the Profinet Certification process to ensure compliance with industrial automation standards. During this process, the use of non-standard protocols such as CDP results in test failures, preventing certification. As a result, Cisco devices intended for Profinet environments must use IEEE standard LLDP instead of CDP.

Enabling PROFINET

To enable PROFINET, follow these steps:

Procedure

Step 1 Enter global configuration mode:

```
Switch# configure terminal
```

Step 2 Enable PROFINET on the switch:

```
Switch(config)# profinet
```

After PROFINET is enabled, you can configure the device as described in Configuring the Switch with STEP7/TIA, on page 2675 (recommended) or by using the Cisco IOS commands provided in the subsequent steps.

Step 3 (Optional) Set the PROFINET device identifier (ID) by using the Cisco IOS software:

```
Switch(config)# profinet id line
```

The maximum length of the ID string can be 240 characters. The only special characters allowed are period (.) and hyphen (-), and they are allowed only in specific positions within the ID string. The ID can have multiple labels within the string. Each label can be from 1 to 63 characters, and labels must be separated by a period (.). The final character in the string must not be zero (0).

For more details about configuring the PROFINET ID, see the PROFINET specification, document number TC2-06-0007a, filename PN-AL-protocol_2722_V22_Oct07, available at PROFIBUS.

This step is optional and can be done through STEP7 or TIA PORTAL STEP 7 or the TIA Portal Automation application installed on the Supervisor (recommended).

Step 4 (Optional) Change the VLAN number. The default VLAN number is 1. The VLAN ID range is from 1 to 4096. One PROFINET VLAN is supported per switch.

```
Switch(config)# profinet vlan vlan_id
```

Note
You must create a VLAN before assigning a new VLAN to PROFINET if you are using a nondefault VLAN.

Step 5 Return to privileged EXEC mode:

```
Switch(config)# end
```

Step 6 Verify your entries:

```
Switch# show running-config
```

Step 7 (Optional) Save your entries in the configuration file:

```
Switch# copy running-config startup-config
```

Configuring the Switch with STEP7/TIA

Complete the following steps to configure the switch with STEP7/TIA. TIA v15.1 is used in the following example. Ensure that you do not use the CLI to configure or modify the switch configuration when PROFINET and TIA are in use.

Procedure

Step 1 Check the availability of the GSD file on the switch. You must use the same version that matches the GSD file bundled with the Cisco IOS release image.

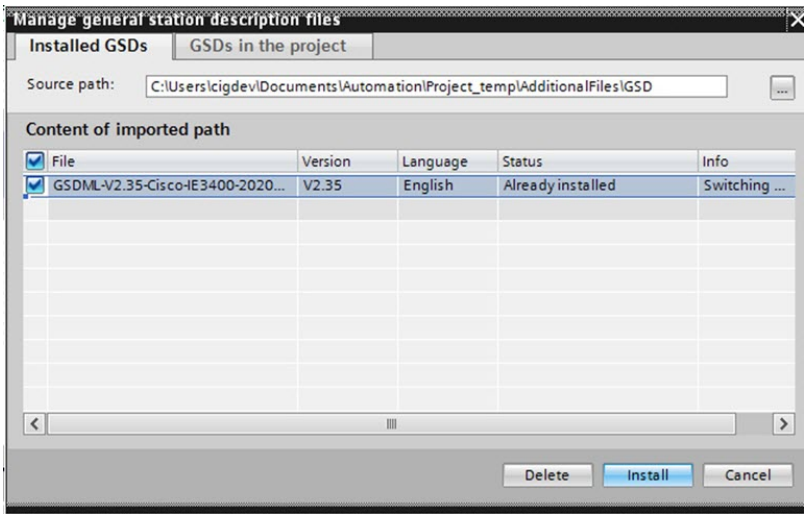
See General Station Description File, on page 2672 for more information.

Step 2 Install the GSD file in STEP7/ TIA:

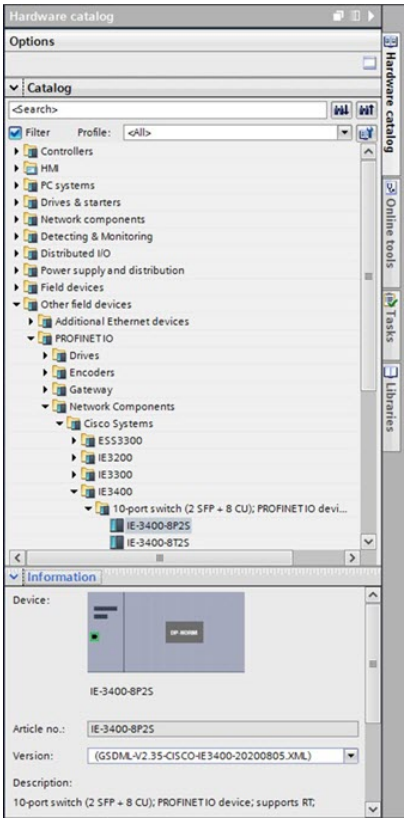
a) In STEP7/TIA, choose **Options** > **Manage general stations description files**, and browse to the location of the GSD file on the PC through source path.

The tool displays all the available GSD files.

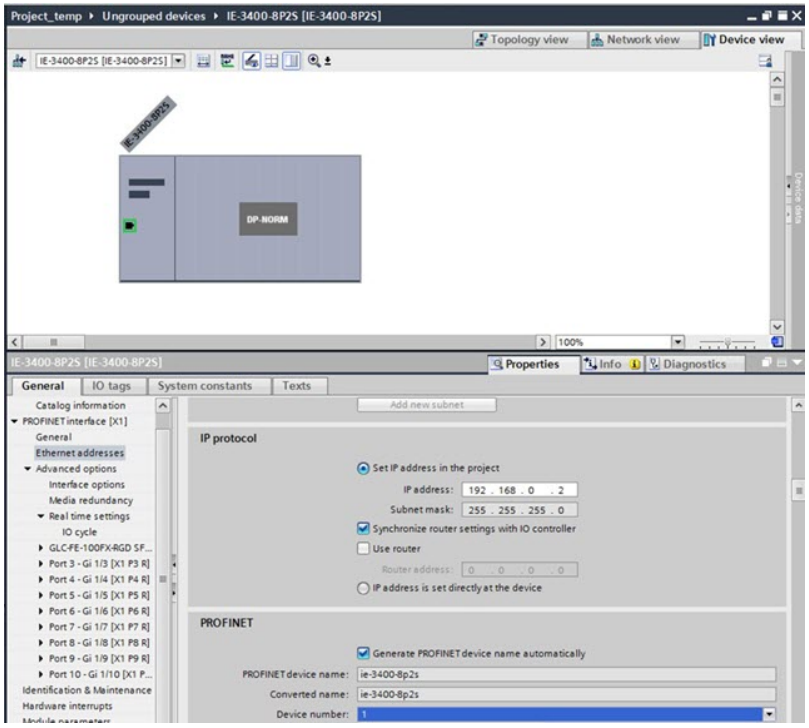
b) Check the check box adjacent to the appropriate the desired GSD file and click **Install**.



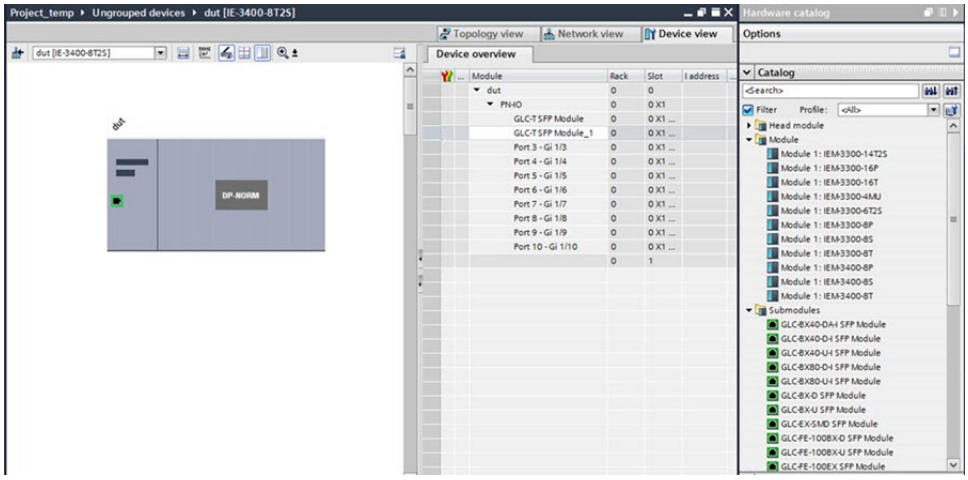
Step 3 After the installation is completed, give it a few seconds to update the **Hardware catalog**. Add the switch from the **Hardware catalog**:



a) In the Device view, configure **IP address** and **PROFINET device name** and save the configuration. These settings are for STEP7/TIA only; the switch is actually configured later during discovery steps.



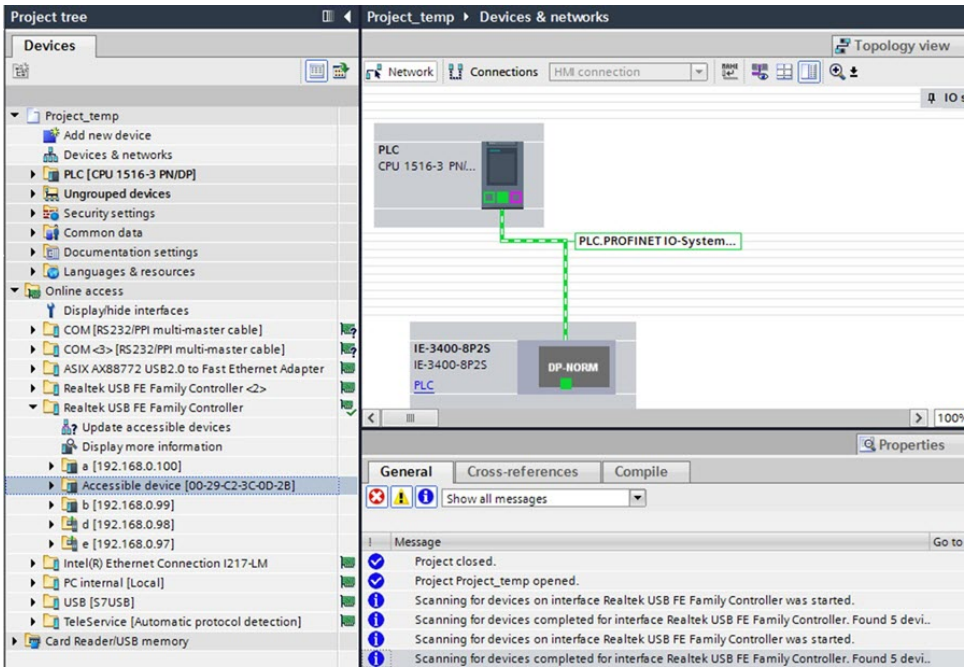
b) Configure the required expansion module or pluggable modules in Device view.



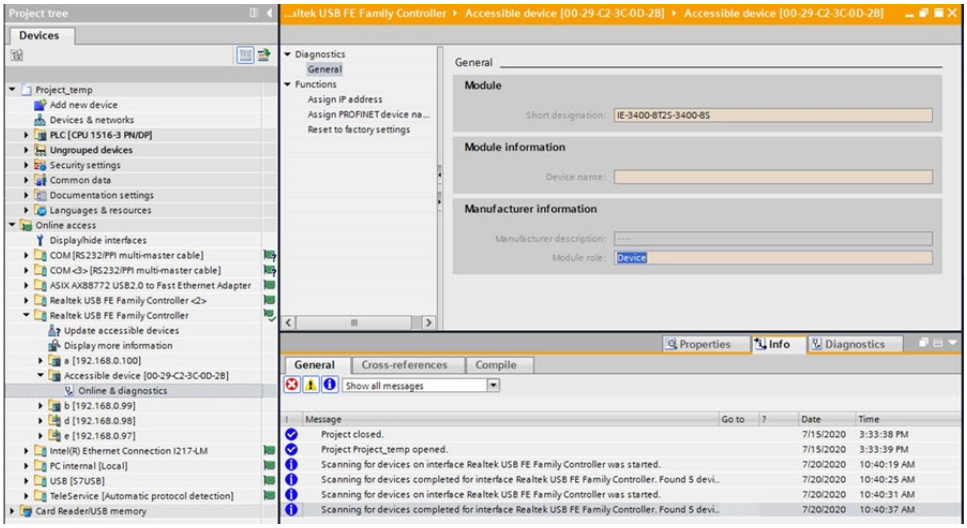
Step 4 After the device is added in the program, discover the device through the interface of the PC connected to the PROFINET topology.

Topology discovery uses LLDP for discovery. LLDP is enabled by default on the switch. You will see the new device listed as **Accessible device** followed by the MAC address under the network card of the PC.

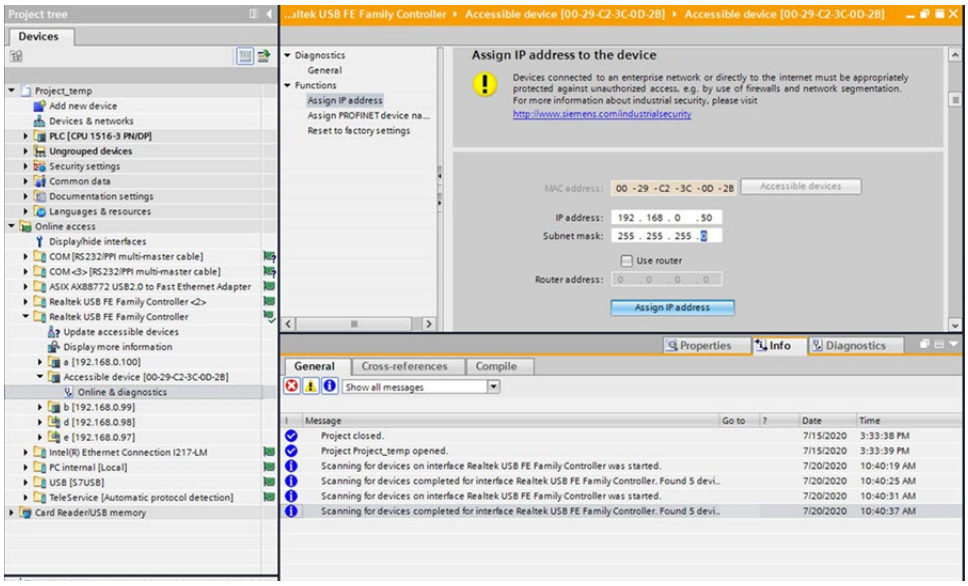
In the **Devices** pane on the left, under **Online access**, find the PC network card and click **Update accessible device**. This initiates the discovery of all the devices in the network.



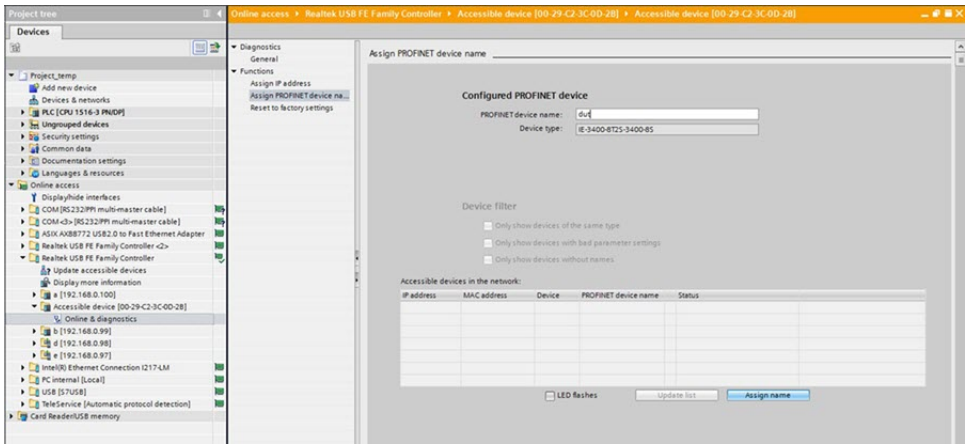
Step 5 In the **Devices** pane, expand the **Accessible device** folder, and click **Online & diagnostics** to further configure the device.



a) In the **IP address** field, enter the IP address and click **Assign IP address** to push the IP address configuration to the switch.

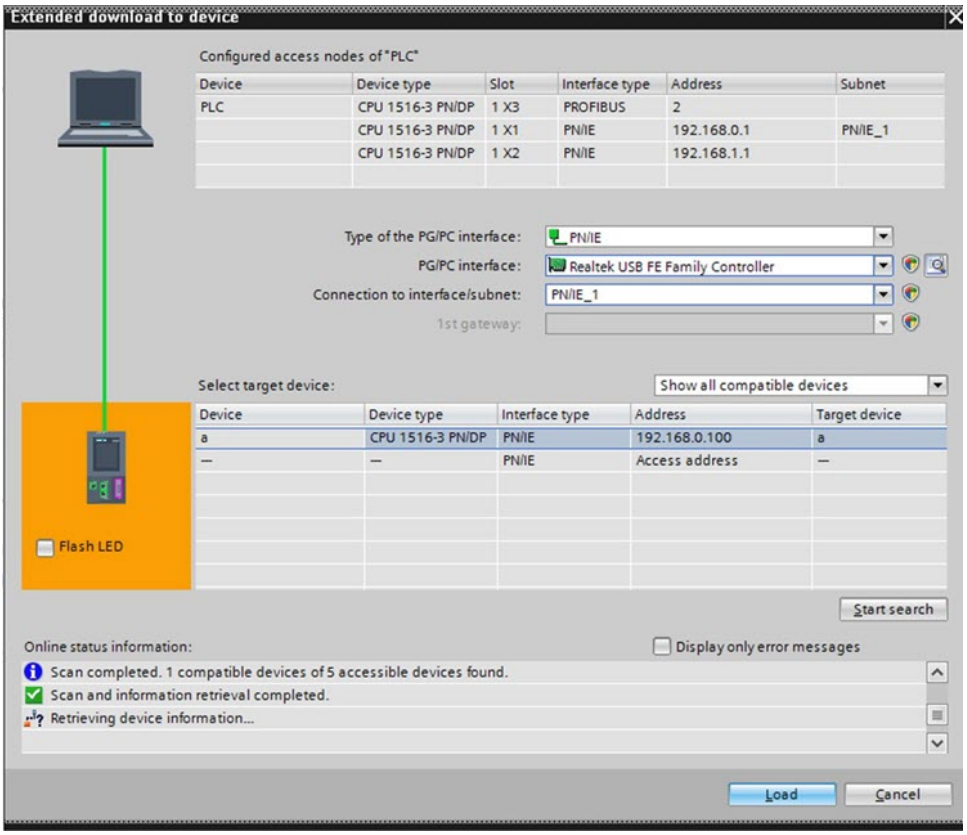


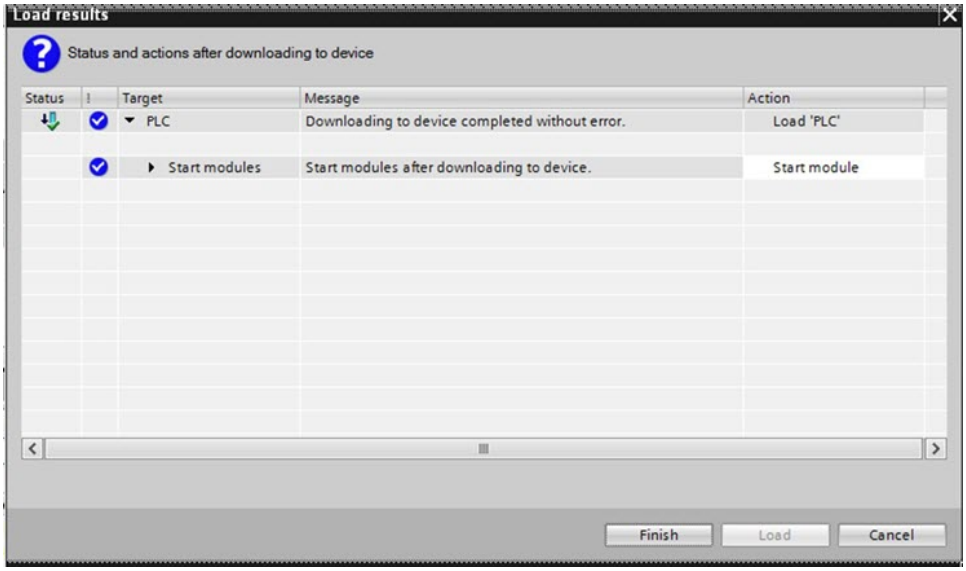
b) In the **PROFINET device name** field, enter the device name and click **Assign name** to push the device name configuration to the switch.



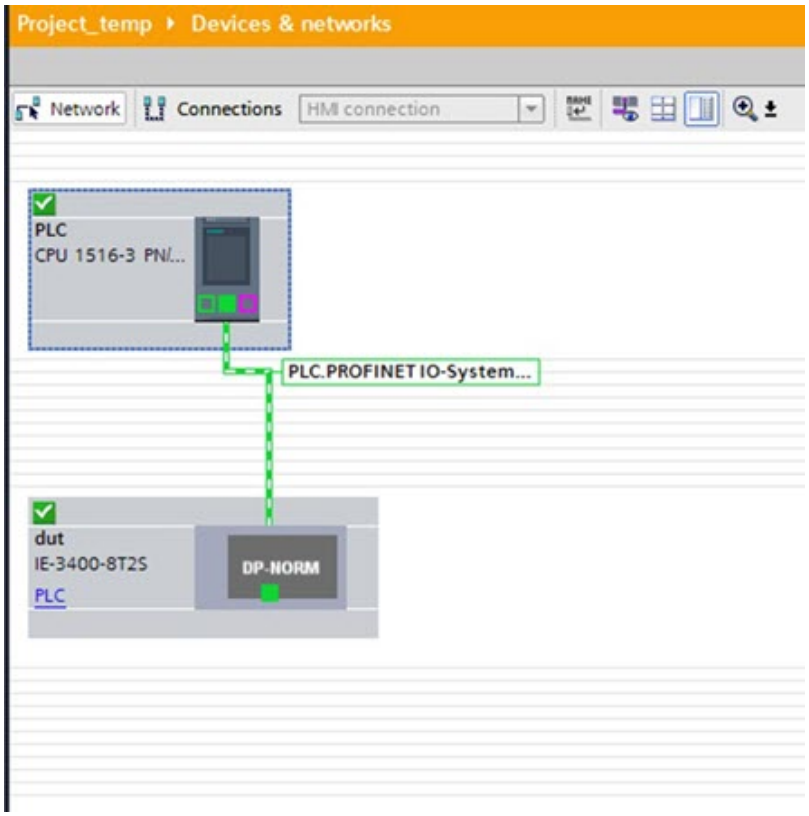
Step 6 Download the project from STEP7/TIA and go online.

a) Compile, download, and load the project to the PLC (I/O Controller).


Extended download to device
Configured access nodes of "PLC"
Device
Device type
Slot
Interface type
Address
Subnet
PLC
CPU 1516-3 PN/DP
1 X3
PROFIBUS
2
CPU 1516-3 PN/DP
1 X1
PN/IE
192.168.0.1
PN/IE_1
CPU 1516-3 PN/DP
1 X2
PN/IE
192.168.1.1
Type of the PG/PC interface:
PN/IE
PG/PC interface:
Realtek USB FE Family Controller
Connection to interface/subnet:
PN/IE_1
1st gateway:
Select target device:
Show all compatible devices
Device
Device type
Interface type
Address
Target device
a
CPU 1516-3 PN/DP
PN/IE
192.168.0.100
a
—
—
PN/IE
Access address
—
Flash LED
Start search
Online status information:
Display only error messages
Scan completed. 1 compatible devices of 5 accessible devices found.
Scan and information retrieval completed.
Retrieving device information...
Load
Cancel


Load results
Status and actions after downloading to device
Status
Target
Message
Action
PLC
Downloading to device completed without error.
Load 'PLC'
Start modules
Start modules after downloading to device.
Start module
Finish
Load
Cancel

b) Go online.



PROFINET Subsystem

The switch enables the PROFINET subsystem. By default, the switch automatically configures its interfaces to manage PROFINET frames with priority tagging. When the IOSd detects PROFINET traffic, it automatically applies the configuration at runtime, using the **switchport voice vlan dot1p** command on each active interface. This runtime configuration ensures the switch receives the priority-tagged PROFINET frames from the LAN.

```
!
interface GigabitEthernet1/1
switchport voice vlan dot1p
!
```

Profinet Connection Configuration

When a Profinet connection/session is established, the network device automatically applies and saves the CLI configurations (including SNMP) given in the table below.

Table 201: CLI for the PROFINET Configuration

CLI	Purpose
snmp-server community private RW **snmp-server community public RO**	Configure two Simple Network Management Protocol (SNMP) community strings, each with a read-write and read-only access.
LLDP **lldp timer 5** **lldp holdtime 20** **lldp run**	Enable the Link Layer Discovery Protocol (LLDP) with a frame transmission interval of 5 seconds and a holdtime of 20 seconds.
CDP **no cdp run**	Disable the Cisco Discovery Protocol (CDP).
Power-supply **power-supply dual**	Set up the device to manage dual power supplies, ensuring it can operate redundantly if one power supply fails.

Preventing Default Gateway and CDP Loss During Reloads and Upgrades

Cisco IE switches have Profinet (PN) enabled by default to facilitate management through the Siemens TIA application portal, similar to other I/O devices, Programmable Logic Controllers (PLCs), and I/O devices within the Local Area Network (LAN).

Upon reloading an Industrial Ethernet (IE) switch operating on Cisco IOS XE 17.x or performing an upgrade from version 16.x, you may experience unexpected network connectivity disruptions. This phenomenon arises because the default gateway configuration fails to persist in the running configuration, resulting in its silent disappearance. Moreover, the Cisco Discovery Protocol (CDP) may also be disabled during this process. To rectify the loss of default gateway and CDP configurations, see the *Recommended Solution* section. While both the default gateway and CDP configurations are retained in the startup configuration, they do not appear in the running configuration post-reload or upgrade.

Technical Evaluation

When the Profinet feature is active, the Profinet subsystem conducts several critical checks to ensure proper configuration:

- Non-Zero Values: The switch IP address configured on the Profinet VLAN, gateway address, and netmask must all be non-zero.
- Subnet Consistency: The IP address and gateway address must reside within the same subnet.
- Uniqueness: The IP address and gateway address must not be identical.

If any of these conditions are not met while the Profinet feature is enabled, the default gateway configuration is removed from the running configuration. Additionally, if you save this incomplete configuration to the startup configuration using the **write mem** command, the erroneous settings will persist through power cycles.

Recommended Solution

To rectify the loss of default gateway and CDP configurations, execute the following steps:

1. Disable Profinet.

   Enter the **no profinet** command to disable the Profinet feature.

2. Reconfigure settings.

   Manually re-enter the CDP and default gateway configurations.

3. Save configuration.

   Use the **write mem** command to save the updated configuration.

4. Verify configuration.

   Optionally, reload the switch to confirm that the configurations are correctly reflected in the running configuration post-reload.

5. Check Profinet status.

   Use the **show profinet status** to ensure that Profinet is disabled.

Monitoring and Maintaining PROFINET

Table 202: Commands for Displaying the PROFINET Configuration

Command	Purpose
show profinet alarm	Displays all the alarms supported by PROFINET.
show profinet lldp	Displays whether LLDP is active or inactive on the ports.
show profinet sessions	Displays the currently connected PROFINET sessions.
show profinet status	Displays the status of the PROFINET subsystem.
Show profinet mrp ring 1	Displays the status of the MRP ring.
show lldp neighbor interface *interface_number* **detail**	Displays information about the adjacent interface.

The following example displays the PROFINET status and currently connected PROFINET sessions.

```
Switch#sh profinet status
Profinet                        : Enabled
Connection Status               : Connected
Vlan                            : 1
Profinet ID                     : dut
```

```
GSD version                     : Match
Reduct Ratio                    : 128
MRP                             : Enabled
MRP License Status              : Active
MRP Max Rings Allowed           : 3

Switch#sh profinet session
Session #1
----------
Connected: Yes
Number Of IO CR's: 2
Number Of DiffModules: 0

Session #2
----------
Connected: No
Number Of IO CR's: 0
Number Of DiffModules: 2

Session #3
----------
Connected: No
Number Of IO CR's: 0
Number Of DiffModules: 0

Session #4
----------
Connected: No
Number Of IO CR's: 0
Number Of DiffModules: 0
*************************
Mode = Standard Mode
```

Monitoring Configuration Changes in PROFINET Sub-Systems

The PROFINET sub-system operates in real-time and adjusts the configuration of the device based on provisions made by remote engineering tools such as TIA Portal or in response to incoming network traffic. These dynamic updates to the running configuration often occur without your awareness. As a result, this can potentially lead to unexpected changes in system behavior.

To enhance user awareness and system transparency, syslog messages are generated whenever modifications occur in critical configuration fields. By monitoring these syslog messages, you can stay informed about real-time changes to the system configuration, ensuring better management and understanding of the PROFINET environment.

The following fields or protocols generate syslogs:

- Link Layer Discovery Protocol (LLDP)
- Cisco Discovery Protocol (CDP)
- Simple Network Management Protocol (SNMP)
- IP address and gateway configurations

Example of the syslog messages:

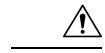
```
*Jun 19 14:41:11.247: %PROFINET_MODULE-6-PN_RUNNING_CONFIG: IP / netmask: persistent
configuration applied
*Jun 19 14:41:11.248: %PROFINET_MODULE-6-PN_RUNNING_CONFIG: Gateway IP criteria met,
configuring default gateway
*Jun 19 14:41:11.260: %SYS-5-CONFIG_I: Configured from console by vty0
```

```
*Jun 19 14:41:11.260: %PROFINET_MODULE-6-PN_RUNNING_CONFIG: CDP Global: service stopped
*Jun 19 14:41:11.276: %SYS-5-CONFIG_I: Configured from console by vty0
*Jun 19 14:41:11.280: %SYS-5-CONFIG_I: Configured from console by vty0
*Jun 19 14:41:11.280: %PROFINET_MODULE-6-PN_RUNNING_CONFIG: SNMP Global: service started
%PROFINET_MODULE-6-PN_RUNNING_CONFIG: LLDP Global: Tx Freq = 5 secs & Holdtime = 20 secs
*Jun 19 14:41:44.283: %PROFINET_MODULE-6-PN_RUNNING_CONFIG: Applying dot1p config on one
or more interfaces
```

Troubleshooting PROFINET

The PLC has LEDs that display red for alarms. The I/O supervisor software monitors those alarms.

To troubleshoot PROFINET, use the **debug profinet** privileged EXEC command with the keywords listed in the following table.



Caution Be aware that the output of a **debug** command might cause a Telnet connection to fail due to long debug outputs. When you use this command, use the serial or console port rather than Telnet using Ethernet to access the Cisco IOS CLI. You should use these commands only under the guidance of a Cisco Technical Support engineer.

Table 203: Commands for Troubleshooting the PROFINET Configuration

Command	Purpose
debug profinet alarm	Displays the alarm status (on or off) and content of the PROFINET alarms.
debug profinet cyclic	Displays information about the time-cycle-based PROFINET Ethernet frames.
debug profinet error	Displays the PROFINET session errors.
debug profinet packet ethernet	Displays information about the PROFINET Ethernet packets.
debug profinet packet udp	Displays information about the PROFINET Upper Layer Data Protocol (UDP) packets.
debug profinet platform	Displays information about the interaction between the Cisco IOS software and PROFINET.
debug profinet topology	Displays the PROFINET topology packets received.
debug profinet trace	Displays a group of traced debug output logs.

CHAPTER 179

Adding SFP Modules to Step 7TIA

Supported Small Form-Factor Pluggables

This document describes how to add Small Form-Factor Pluggable (SFP) modules to the SIMATIC STEP7 or TIA Portal automation applications to enable these applications to recognize the SFP modules in the PROFINET environment.

The following SFPs are supported for the switch and the fiber ports expansion module and can be added to the STEP 7 or TIA Portal Automation application installed on the supervisor:

- GLC-T
- GLC-TE
- GLC-T-RGD
- GLC-FE-100FX-RGD
- GLC-FE-100LX-RGD
- GLC-SX-MM-RGD
- GLC-LX-SM-RGD
- GLC-FE-100FX
- GLC-FE-100LX
- GLC-SX-MMD
- GLC-LH-SMD
- GLC-SX-MM
- GLC-LH-SM

Adding SFPs to the Hardware Configuration in SS7/TIA

Follow this procedure to add SFPs to the hardware configuration in STEP7 or TIA.

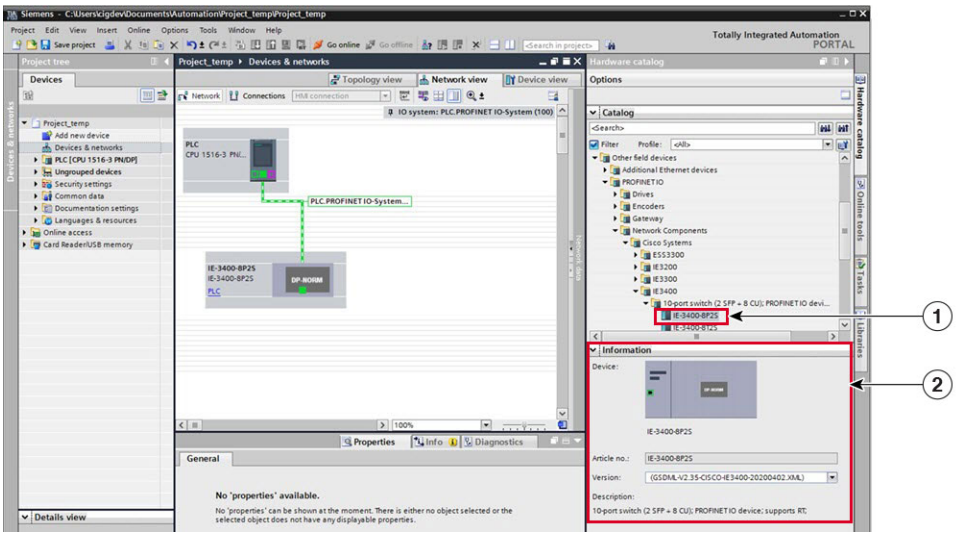
Before you begin

Install the latest GSD file for your IE switches in STEP7 or TIA (see Configuring the Switch with STEP7/TIA, on page 2675 for more information). The GSD file is not backward compatible. Combination ports do not have defaults and must be re-created in STEP7 or TIA with the new GSD file.

Procedure

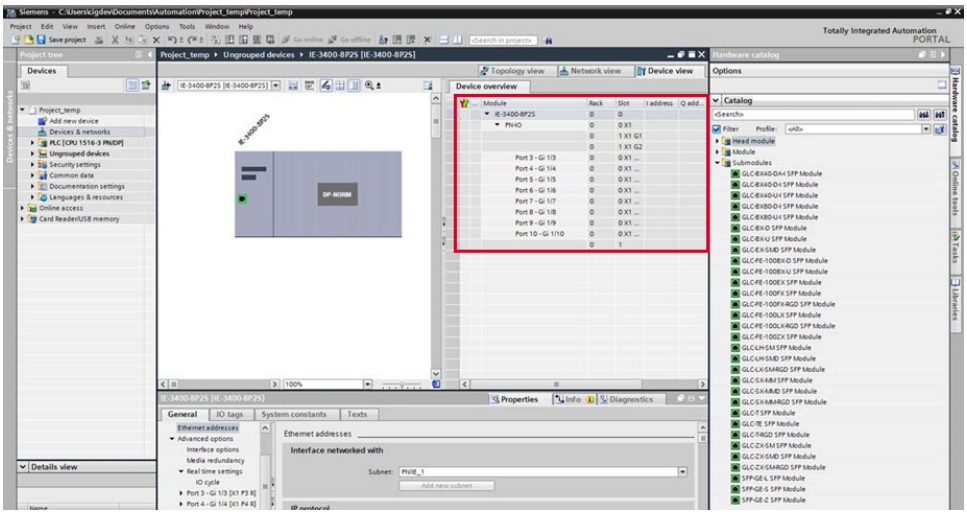
Step 1 In STEP7 or TIA, open the project containing the Cisco IE switch, and double-click **Devices & networks** in the **Project tree**.

Step 2 From the **Hardware catalog** in the devices and network editor, select the Device Access Point (DAP) name for your switch.



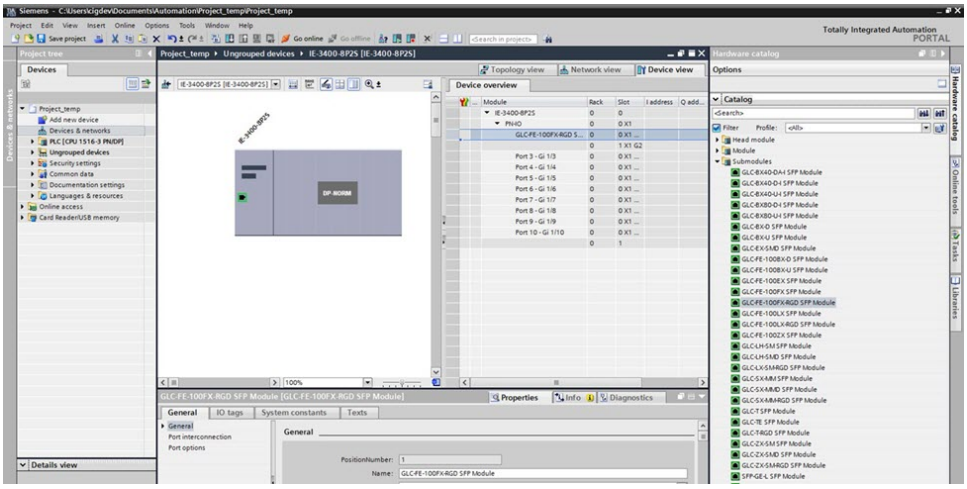
Step 3 Double-click the DAP name to go to **Device view** tab (see callout 1).

Step 4 Expand the PN-IO row (see callout 2) in Device overview section if the port list is not displayed.



Step 5 Select the modules that you have installed in the switch from the **Hardware catalog** and drag them to the appropriate rows in the table area of **Device view**: Appropriate SFP module for port type (combo port or fiber-only port)

The following figure shows the **Device view** and **Hardware catalog** with SFP modules. There is an uplink module for Gi SFP. The device in TIA is now ready to compile and download.



The following figure shows the **Device view** and **Hardware catalog** with SFP modules for IE3400 with the Fiber expansion module that has downlink SFP port Gi2/1-8, and where SFP GLC-FE-100FX-RGD is dragged to uplink port Gi2/8. The device in TIA is now ready to compile and download.