System Management Configuration Guide, Cisco IOS XE 17.18.x (Catalyst 9200 Switches)

First Published: 2025-08-08

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

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: https://www.cisco.com/c/en/us/about/legal/trademarks.html. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2025 Cisco Systems, Inc. All rights reserved.

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 , on page iii
- Related Documentation, on page v
- Obtaining Documentation and Submitting a Service Request, on page v

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.

Convention	Description
{x | y}	Required alternative keywords are grouped in braces and separated by vertical bars.
[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 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

Related Documentation

Note Before installing or upgrading the device, refer to the device release notes.

- Cisco Catalyst 9400 Series Switches documentation, located at:

  http://www.cisco.com/go/c9400

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS version 2.0.

CONTENTS

PREFACE

Preface iii

Document Conventions iii

Related Documentation v

Obtaining Documentation and Submitting a Service Request v

CHAPTER 1

Administering the Device 1

Information About Administering the Device 1

System Time and Date Management 1

System Clock 1

Network Time Protocol 2

NTP Implementation 6

DNS 7

Default DNS Settings 7

Login Banners 7

Default Banner Configuration 7

MAC Address Table 8

MAC Address Table Creation 8

MAC Addresses and VLANs 8

Default MAC Address Table Settings 8

ARP Table Management 9

How to Administer the Device 9

Configuring the Time and Date Manually 9

Setting the System Clock 9

Configuring the Time Zone 10

Configuring Summer Time (Daylight Saving Time) 11

Configuring NTP 13

Default NTP Configuration 13
Configuring NTP Authentication 13
Configuring Poll-Based NTP Associations 15
Configuring Broadcast-Based NTP Associations 17
Configuring NTP Access Restrictions 18
Configuring a System Name 20
Setting Up DNS 21
Configuring a Message-of-the-Day Login Banner 23
Configuring a Login Banner 24
Managing the MAC Address Table 25
Changing the Address Aging Time 25
Configuring MAC Address Change Notification Traps 26
Configuring MAC Address Move Notification Traps 29
Configuring MAC Threshold Notification Traps 31
Disabling MAC Address Learning on VLAN 33
Adding and Removing Static Address Entries 34
Configuring Unicast MAC Address Filtering 35
Monitoring and Maintaining Administration of the Device 36
Configuration Examples for Device Administration 37
Example: Setting the System Clock 37
Examples: Configuring Summer Time 37
Example: Configuring a MOTD Banner 37
Example: Configuring a Login Banner 38
Example: Configuring MAC Address Change Notification Traps 38
Example: Configuring MAC Threshold Notification Traps 38
Example: Adding the Static Address to the MAC Address Table 39
Example: Configuring Unicast MAC Address Filtering 39
Additional References for Device Administration 39
Feature History for Device Administration 39

CHAPTER 2 **Boot Integrity Visibility** 41
Information About Boot Integrity Visibility 41
Image Signing and Bootup 41
Verifying the Software Image and Hardware 42

Verifying Platform Identity and Software Integrity 43
Verifying Image Signing 46
Additional References for Boot Integrity Visibility 47
Feature History for Boot Integrity Visibility 47

CHAPTER 3 **Performing Device Setup Configuration 49**
Restrictions for Performing Device Setup Configuration 49
Information About Performing Device Setup Configuration 49
Device Boot Process 49
Software Install Overview 50
Software Boot Modes 51
Installing the Software Package 52
Terminating a Software Install 52
Devices Information Assignment 52
Default Switch Information 53
DHCP-Based Autoconfiguration Overview 53
DHCP Client Request Process 54
DHCP-Based Autoconfiguration and Image Update 55
Restrictions for DHCP-Based Autoconfiguration 55
DHCP Autoconfiguration 55
DHCP Auto-Image Update 55
DHCP Server Configuration Guidelines 56
Purpose of the TFTP Server 56
Purpose of the DNS Server 57
How to Obtain Configuration Files 57
How to Control Environment Variables 58
Scheduled Reload of the Software Image 58
How to Perform Device Setup Configuration 59
Configuring DHCP Autoconfiguration (Only Configuration File) 59
Manually Assigning IP Information to Multiple SVIs 61
Modifying Device Startup Configuration 62
Specifying a Filename to Read and Write a System Configuration 63
Booting the Device in Installed Mode 64
Booting a Device in Bundle Mode 66

Configuring a Scheduled Software Image Reload 66
Configuration Examples for Device Setup Configuration 67
Examples: Displaying Software Bootup in Install Mode 67
Example: Managing an Update Package 71
Verifying Software Install 73
Example: Configuring a Device to Download Configurations from a DHCP Server 74
Example: Scheduling Software Image Reload 74
Additional References For Performing Device Setup 75
Feature History for Performing Device Setup Configuration 75

CHAPTER 4 **Configuring Application Visibility and Control in a Wired Network 77**
Information About Application Visibility and Control in a Wired Network 77
Supported AVC Class Map and Policy Map Formats 77
Restrictions for Wired Application Visibility and Control 79
How to Configure Application Visibility and Control 81
Configuring Application Visibility and Control in a Wired Network 81
Enabling Application Recognition on an interface 81
Creating AVC QoS Policy 82
Applying a QoS Policy to the switch port 84
Configuring Wired AVC Flexible Netflow 85
NBAR2 Custom Applications 101
NBAR2 Dynamic Hitless Protocol Pack Upgrade 104
Monitoring Application Visibility and Control 106
Examples: Application Visibility and Control Configuration 106
Basic Troubleshooting - Questions and Answers 118
Additional References for Application Visibility and Control 119
Feature History for Application Visibility and Control in a Wired Network 120

CHAPTER 5 **Configuring SDM Templates 121**
Information About SDM Templates 121
How to Configure SDM Templates 121
Setting the SDM Template 121
Monitoring and Maintaining SDM Templates 122
Configuration Examples for SDM Templates 123

Examples: Displaying SDM Templates 123
Additional References for SDM Templates 124
Feature History for SDM Templates 124

CHAPTER 6 **Configuring System Message Logs** 127
Information About Configuring System Message Logs 127
System Messsage Logging 127
System Log Message Format 128
Default System Message Logging Settings 129
Syslog Message Limits 129
How to Configure System Message Logs 130
Setting the Message Display Destination Device 130
Synchronizing Log Messages 131
Disabling Message Logging 132
Enabling and Disabling Time Stamps on Log Messages 133
Enabling and Disabling Sequence Numbers in Log Messages 134
Defining the Message Severity Level 135
Limiting Syslog Messages Sent to the History Table and to SNMP 135
Logging Messages to a UNIX Syslog Daemon 136
Monitoring and Maintaining System Message Logs 137
Monitoring Configuration Archive Logs 137
Configuration Examples for System Message Logs 137
Example: Stacking System Message 137
Example: Switch System Message 138
Additional References for System Message Logs 138
Feature History for System Message Logs 138

CHAPTER 7 **Configuring Online Diagnostics** 141
Restrictions for Online Diagnostics 141
Information About Configuring Online Diagnostics 141
Generic Online Diagnostics (GOLD) Tests 142
How to Configure Online Diagnostics 145
Starting Online Diagnostic Tests 145
Configuring Online Diagnostics 145

Scheduling Online Diagnostics 146
Configuring Health-Monitoring Diagnostics 147
Monitoring and Maintaining Online Diagnostics 149
Configuration Examples for Online Diagnostics 150
Examples: Start Diagnostic Tests 150
Example: Configure a Health-Monitoring Test 150
Example: Schedule Diagnostic Test 150
Example: Displaying Online Diagnostics 150
Additional References for Online Diagnostics 152
Feature History for Configuring Online Diagnostics 152

CHAPTER 8 **Consistency Checker** 153
Limitations for Consistency Checker 153
Information about Consistency Checker 154
Running the Consistency Checker 155
Output Examples for Consistency Checker 155
Feature History for Consistency Checker 161

CHAPTER 9 **Managing Configuration Files** 163
Prerequisites for Managing Configuration Files 163
Restrictions for Managing Configuration Files 163
Information About Managing Configuration Files 163
Types of Configuration Files 163
Configuration Mode and Selecting a Configuration Source 164
Configuration File Changes Using the CLI 164
Location of Configuration Files 164
Copy Configuration Files from a Network Server to the Device 165
Copying a Configuration File from the Device to a TFTP Server 165
Copying a Configuration File from the Device to an RCP Server 166
Copying a Configuration File from the Device to an FTP Server 167
Copying files through a VRF 168
Copy Configuration Files from a Switch to Another Switch 168
Configuration Files Larger than NVRAM 169
Configuring the Device to Download Configuration Files 169

How to Manage Configuration File Information 170
Displaying Configuration File Information 170
Modifying the Configuration File 171
Copying a Configuration File from the Device to a TFTP Server 172
What to Do Next 173
Copying a Configuration File from the Device to an RCP Server 173
Examples 174
What to Do Next 175
Copying a Configuration File from the Device to the FTP Server 175
Examples 176
What to Do Next 177
Copying a Configuration File from a TFTP Server to the Device 177
What to Do Next 178
Copying a Configuration File from the rcp Server to the Device 178
Examples 179
What to Do Next 179
Copying a Configuration File from an FTP Server to the Device 179
Examples 180
What to Do Next 181
Maintaining Configuration Files Larger than NVRAM 181
Compressing the Configuration File 181
Storing the Configuration in Flash Memory on Class A Flash File Systems 183
Loading the Configuration Commands from the Network 184
Copying Configuration Files from Flash Memory to the Startup or Running Configuration 185
Copying Configuration Files Between Flash Memory File Systems 186
Copying a Configuration File from an FTP Server to Flash Memory Devices 187
What to Do Next 188
Copying a Configuration File from an RCP Server to Flash Memory Devices 188
Copying a Configuration File from a TFTP Server to Flash Memory Devices 189
Re-executing the Configuration Commands in the Startup Configuration File 190
Clearing the Startup Configuration 190
Deleting a Specified Configuration File 191
Specifying the CONFIG_FILE Environment Variable on Class A Flash File Systems 192
What to Do Next 193

Configuring the Device to Download Configuration Files 194
Configuring the Device to Download the Network Configuration File 194
Configuring the Device to Download the Host Configuration File 195
Feature History for Managing Configuration Files 197

CHAPTER 10 **Secure Copy** 199
Prerequisites for Secure Copy 199
Information About Secure Copy 199
Secure Copy Performance Improvements 200
How to Configure Secure Copy 200
Configuring Secure Copy 200
Configuring SCP Username Password 201
Enabling Secure Copy on the SSH Server 202
Configuration Examples for Secure Copy 204
Example: Secure Copy Configuration Using Local Authentication 204
Example: Secure Copy Server-Side Configuration Using Network-Based Authentication 204
Additional References for Secure Copy 204
Feature History for Secure Copy 205

CHAPTER 11 **Configuration Replace and Configuration Rollback** 207
Prerequisites for Configuration Replace and Configuration Rollback 207
Restrictions for Configuration Replace and Configuration Rollback 208
Information About Configuration Replace and Configuration Rollback 208
Configuration Archive 208
Configuration Replace 209
Configuration Rollback 210
Configuration Rollback Confirmed Change 210
Benefits of Configuration Replace and Configuration Rollback 210
How to Use Configuration Replace and Configuration Rollback 211
Creating a Configuration Archive 211
Performing a Configuration Replace or Configuration Rollback Operation 212
Monitoring and Troubleshooting the Feature 215
Configuration Examples for Configuration Replace and Configuration Rollback 217
Creating a Configuration Archive 217

Replacing the Current Running Configuration with a Saved Cisco IOS Configuration File 217
Reverting to the Startup Configuration File 218
Performing a Configuration Replace Operation with the configure confirm Command 218
Performing a Configuration Rollback Operation 218
Additional References for Configuration Replace and Configuration Rollback 220
Feature History for Configuration Replace and Configuration Rollback 220

CHAPTER 12 Software Maintenance Upgrade 221
Restrictions for Software Maintenance Upgrade 221
Information About Software Maintenance Upgrade 221
SMU Overview 221
SMU Workflow 222
SMU Package 222
SMU Reload 222
How to Manage Software Maintenance Updates 222
Installing an SMU Package: 1-Step Process 223
Installing an SMU Package: 3-Step Process 224
Managing an SMU 225
Configuration Examples for Software Maintenance Upgrade 225
Additional References for Software Maintenance Upgrade 238
Feature History for Software Maintenance Upgrade 238

CHAPTER 13 Working with the Flash File System 239
Information About the Flash File System 239
Displaying Available File Systems 239
Setting the Default File System 242
Displaying Information About Files on a File System 242
Changing Directories and Displaying the Working Directory 243
Creating Directories 244
Removing Directories 244
Copying Files 244
Copying Files from One Device in a Stack to Another Device in the Same Stack 245
Deleting Files 246
Creating, Displaying and Extracting Files 247

Additional References for Flash File System 249
Feature History for Flash File System 249

CHAPTER 14 Performing Factory Reset 251
Prerequisites for Performing a Factory Reset 251
Restrictions for Performing a Factory Reset 251
Information About Performing a Factory Reset 251
Secure Data Wipe 252
How to Perform a Factory Reset 253
Configuration Examples for Performing a Factory Reset 254
Additional References for Performing a Factory Reset 256
Feature History for Performing a Factory Reset 256

CHAPTER 15 Configuring Secure Storage 259
Information About Secure Storage 259
Enabling Secure Storage 259
Disabling Secure Storage 260
Verifying the Status of Encryption 260
Feature History for Secure Storage 261

CHAPTER 16 Trace Management 263
Information About Trace Management 263
Introduction to Binary Tracing 263
Introduction to Conditional Debugging and Radioactive Tracing 263
Tracing Levels 264
Payload Filter 265
How to Configure Conditional Debugging 266
Conditional Debugging and Radioactive Tracing 266
Configuring Conditional Debugging 266
Collecting Trace Files 268
Copying Archived Trace Files 268
Configuring Payload Filter 269
Configuration Examples for Trace Management 269
Additional References for Trace Management 272

Feature History for Trace Management 272

CHAPTER 17 Consent Token 273

Restrictions for Consent Token 273

Information About Consent Token 273

Consent Token Authorization Process for System Shell Access 274

Feature History for Consent Token 275

CHAPTER 18 Troubleshooting the Software Configuration 277

Information About Troubleshooting the Software Configuration 277

Software Failure on a Switch 277

Lost or Forgotten Password on a Device 277

Ping 278

Layer 2 Traceroute 278

Layer 2 Traceroute Guidelines 278

IP Traceroute 279

Debug Commands 280

System Report 280

Onboard Failure Logging on the Switch 282

Possible Symptoms of High CPU Utilization 282

How to Troubleshoot the Software Configuration 283

Booting from the Recovery Partition 283

Recovering from a Lost or Forgotten Password 283

Procedure with Password Recovery Enabled 284

Procedure with Password Recovery Disabled 286

Preventing Autonegotiation Mismatches 288

Troubleshooting SFP Module Security and Identification 288

Executing Ping 289

Monitoring Temperature 289

Monitoring the Physical Path 289

Executing IP Traceroute 289

Redirecting Debug and Error Message Output 290

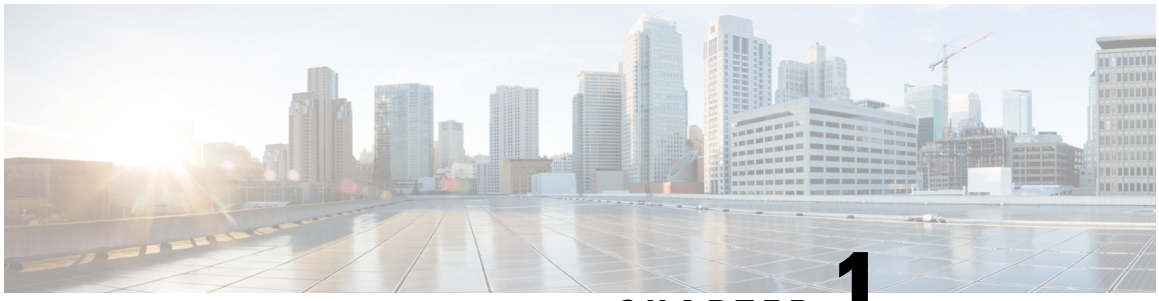
Using the show platform Command 290

Using the show debug command 290

Verifying Troubleshooting of the Software Configuration 291
Displaying OBFL Information 291
Example: Verifying the Problem and Cause for High CPU Utilization 291
Scenarios for Troubleshooting the Software Configuration 292
Scenarios to Troubleshoot Power over Ethernet (PoE) 292
Configuration Examples for Troubleshooting Software 294
Example: Pinging an IP Host 294
Example: Performing a Traceroute to an IP Host 295
Additional References for Troubleshooting Software Configuration 296
Feature History for Troubleshooting Software Configuration 296

CHAPTER 19 **Line Auto Consolidation** 297
Line Auto Consolidation 297
Feature History for Line Auto Consolidation 303

CHAPTER 20 **Troubleshooting System Management** 305
Overview 305
Support Articles 305
Feedback Request (Reference) 306
Disclaimer and Caution (Reference) 307



CHAPTER 1

Administering the Device

- Information About Administering the Device, on page 1
- How to Administer the Device, on page 9
- Configuration Examples for Device Administration, on page 37
- Additional References for Device Administration, on page 39
- Feature History for Device Administration, on page 39

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.

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.

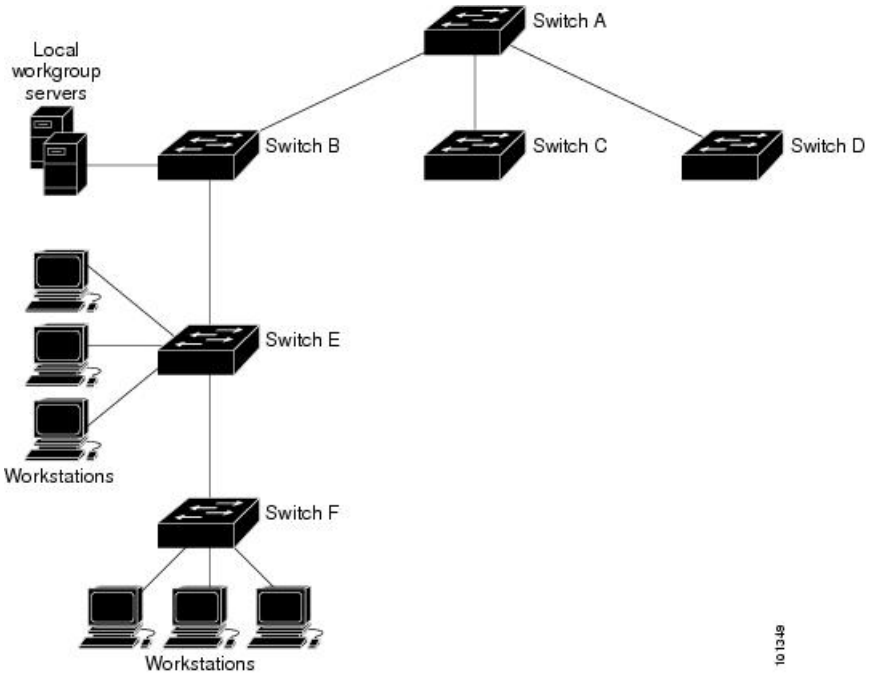
The time kept on a device is a critical resource; you should use the security features of NTP to avoid the accidental or malicious setting of an incorrect time. Two mechanisms are available: an access list-based restriction scheme and an encrypted authentication mechanism.

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 1: 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 Cisco Catalyst 9000 Series 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 1: 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 2: Default Settings for the MAC Address

Feature	Default Setting
Aging time	300 seconds

Feature	Default Setting
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.

	Command or Action	Purpose
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. • *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

	Command or Action	Purpose
		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 Summer Time (Daylight Saving Time)

To configure summer time (daylight saving time) in areas where it starts and ends on a particular day of the week each year, 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	**clock summer-time** *zone* **date** *date month year hh:mm date month year hh:mm* [*offset*]] **Example:** `Device(config)# clock summer-time PDT`	Configures summer time to start and end on specified days every year.

	Command or Action	Purpose
	`date` `10 March 2013 2:00 3 November 2013 2:00`	
Step 4	**clock summer-time** *zone* **recurring** [*week day month hh:mm week day month hh:mm* [*offset*]] **Example:** `Device(config)# clock summer-time PDT recurring 10 March 2013 2:00 3 November 2013 2:00`	Configures summer time to start and end on the specified days every year. All times are relative to the local time zone. The start time is relative to standard time. The end time is relative to summer time. Summer time is disabled by default. If you specify **clock summer-time** *zone* **recurring** without parameters, the summer time rules default to the United States rules. If the starting month is after the ending month, the system assumes that you are in the southern hemisphere. • *zone*—Specifies the name of the time zone (for example, PDT) to be displayed when summer time is in effect. • (Optional) *week*— Specifies the week of the month (1 to 4, **first**, or **last**). • (Optional) *day*—Specifies the day of the week (Sunday, Monday...). • (Optional) *month*—Specifies the month (January, February...). • (Optional) *hh:mm*—Specifies the time (24-hour format) in hours and minutes. • (Optional) *offset*—Specifies the number of minutes to add during summer time. The default is 60.
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`	

Configuring NTP

These following sections provide configuration information on NTP:

Default NTP Configuration

Note Certain device configurations and protocols are identified as insecure and pose a security risk to Cisco devices. While existing deployments continue to function, you must intentionally enable these features for new installations. For more information on remediation, refer to Cisco C9000 Switching IOSXE – Resilient Infrastructure Playbook.

shows the default NTP configuration.

Table 3: 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.
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.

	Command or Action	Purpose
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 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:**	Allows the software clock to be synchronized by an NTP time server.

	Command or Action	Purpose
	`Device(config)# ntp server 172.16.22.44 key 42`	• *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:** `Device> 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 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 3. By default, version 3 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 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 3. By default, version 3 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 clock hop among peers. Use the **no** form of this command to remove a server association.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

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

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 gigabitethernet1/0/1`	Configures an interface and enters interface configuration mode.
Step 4	[**no**] **ntp broadcast** [**version** *number*] [**key** *key-id*] [*destination-address*] **Example:** `Device(config-if)# ntp broadcast version 2`	Enables the interface to send NTP broadcast packets to a peer. • *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.

	Command or Action	Purpose
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:**	Create an access group, and apply a basic IP access list.. • **query-only**: NTP control queries.

	Command or Action	Purpose
	`Device(config)# ntp access-group peer 99`	• **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 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 gigabitethernet1/0/1`	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:** `Device(config-if)# end`	Returns to privileged EXEC mode.

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.

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

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). **Note** Certain device configurations and protocols are identified as insecure and pose a security risk to Cisco devices. While existing deployments continue to function, you must intentionally enable these features for new installations. For more information on remediation, refer to Cisco C9000 Switching IOS XE – Resilient Infrastructure Playbook.
Step 4	**ip name-server** *server-address1* [*server-address2 ... server-address6*] **Example:** `Device(config)# ip name-server 192.168.1.100 192.168.1.200 192.168.1.300`	Specifies the address of one or more name servers to use for name and address resolution. 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.

	Command or Action	Purpose
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** **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	**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:**	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	**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:** `Device(config)# snmp-server enable traps mac-notification change`	Enables the device to send MAC address change notification traps to the NMS.

	Command or Action	Purpose
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 gigabitethernet1/0/2`	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:** `Device# show running-config`	Verifies your entries.

	Command or Action	Purpose
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 the **snmp-server host** command, we recommend that you define this string by using the **snmp-server community**

	Command or Action	Purpose
		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 1/0/1`	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 gigabitethernet1/2/1
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 gigabitethernet1/1/1
```

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
```

Additional References for Device Administration

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9200 Series Switches)*
For configuring VRF-aware services for NTP.	*Configuring Multi-VRF CE in IP Routing Configuration Guide*

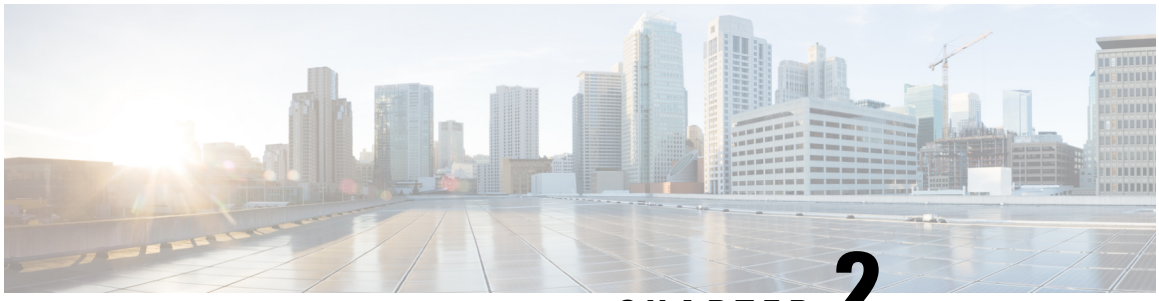
Feature History for Device Administration

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	Device Administration	The device administration allows to configure the system time and date, system name, a login banner, and set up the DNS.
Cisco IOS XE Cupertino 17.9.1	Active VLAN Support	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 2

Boot Integrity Visibility

- Information About Boot Integrity Visibility, on page 41
- Verifying the Software Image and Hardware, on page 42
- Verifying Platform Identity and Software Integrity, on page 43
- Verifying Image Signing, on page 46
- Additional References for Boot Integrity Visibility, on page 47
- Feature History for Boot Integrity Visibility, on page 47

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 Catalyst 9000 Series 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.

Catalyst 9000 Series 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:** `Device# show platform sudi certificate sign nonce 123`	Displays checksum record for the specific SUDI. • (Optional) **sign** - Show signature • (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
aXBJRD1RRGx6T0FZUHQwRTJURVFFQUFjQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
PTANBgkqhkiG9w0BAQsFAAOCAQEAgLUxZfNmrXZ6ZMGX69dDPkmvp9cFqXR538LF
PdypCRuSk20GF8OeDUOsuIi4mbB87JSOWvLomdBtXdnxzRu4kPZNFz/7pjAVRT3R
gwMMyiEnDWQSvy7e4SZmyVgej55e3hTW/LTeU8lCE0KRoYGDce5Phv2zdHtIsXrV
XsY+Fropfnttl1FV9qqDskDWcKf0bos6VsyWUpSCEGqF7LfNnBTKYvXUUmkXHKf/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:C9200L-24T-4G SN:FDO1946BG05/O=Cisco/OU=ACT-2 Lite
SUDI/CN=C9200L-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 cat9k_iosxe.16.10.01.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: C9200L-24T-4G
Boot 0 Version: SBOOT0.v27
Boot 0 Hash:
EE98DCD0D6AEA85C8891039F649664FCC3CF709CCFC7A6F248C9D5BA8463528F
Boot Loader Version: System Bootstrap, Version 10.2, DEVELOPMENT SOFTWARE
Boot Loader Hash:
922DB87E7A153A79EE9AE37311A1FDE2313C999F21032F8A1E7EF4935D3E742765 7E4CDEE537E7B3C50E84121C00BD2556786A4EE155D8C0AFF67F63F1A69B
OS Version: 16.10.01
OS Hashes:
cat9k_lite-rpbase.16.10.01.SPA.pkg :
DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0
cat9k_lite-rpboot.16.10.01.SPA.pkg :
AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057
cat9k_lite-srdriver.16.10.01.SPA.pkg :
4EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057AD6A9673E211
cat9k_lite-webui.16.10.01.SPA.pkg :
CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057AD6A9673E2114EA7
cat9k-wlc.16.10.01.SPA.pkg :
AA7ED0AE935CB0ED84E0DDD155C1DEFDB03EB0C64057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DDD155C1DEFB03EB0C64057AD6A9673E2114EA7CCCA
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: C9200L-24T-4G
Boot 0 Version: SBOOT0.v27
Boot 0 Hash:
EE98DCD0D6AEA85C8891039F649664FCC3CF709CCFC7A6F248C9D5BA8463528F
Boot Loader Version: System Bootstrap, Version 10.2, DEVELOPMENT SOFTWARE
Boot Loader Hash:
922DB87E7A153A79EE9AE37311A1FDE2313C999F21032F8A1E7EF4935D3E742765 7E4CDEE537E7B3C50E84121C00BD2556786A4EE155D8C0AFF67F63F1A69B
OS Version: 16.10.01
```

```
OS Hashes:
cat9k_lite_iosxe.16.10.01.SPA.bin :
F4CAD08EE1EF841C9A2E3ED854082F08F3CEA9336F38E45669D4D8E15AD15E365B922AC8B4D0CD5B63E2806D6A1EDAB7839DD9D08CD7E366A49ED648C113440
cat9k_lite-rpbase.16.10.01.SPA.pkg :
DDD155C1DEFDB03EB064057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DD155C1DEFDB03EB064057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0
cat9k_lite-rpboot.16.10.01.SPA.pkg :
AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DD155C1DEFDB03EB064057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DD155C1DEFDB03EB064057
cat9k_lite-srdriver.16.10.01.SPA.pkg :
4EA7CCCAAA7ED0AE935CB0ED84E0DD155C1DEFDB03EB064057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DD155C1DEFDB03EB064057AD6A9673E211
cat9k_lite-webui.16.10.01.SPA.pkg :
CCCAAA7ED0AE935CB0ED84E0DD155C1DEFDB03EB064057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DD155C1DEFDB03EB064057AD6A9673E2114EA7
cat9k-wlc.16.10.01.SPA.pkg :
AA7ED0AE935CB0ED84E0DD155C1DEFDB03EB064057AD6A9673E2114EA7CCCAAA7ED0AE935CB0ED84E0DD155C1DEFDB03EB064057AD6A9673E2114EA7CCCA
PCR0: 750E5D2EDAE6E3A68050638E0BFD8619BE4EA13066025D39DF79408719F5177E
PCR8: EB6E739A63F53E703B6CDAF3F6188833CEF6D32E2F726006B9AA34E1E73048C4
Signature version: 1
Signature:
```

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: cat9k-rpboot.17.02.01.SSA.pkg
```

```
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: 494C495459000000000000090000000B - 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: 74396B5F5459504500000009000000018 - 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_cat9k_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=$$
TLV: T=9, L=17, V=CW_FAMILY=$cat9k$
TLV: T=9, L=74, V=CW_IMAGE=$cat9k-rpboot.17.02.01.SSA.pkg$
TLV: T=9, L=20, V=CW_VERSION=$17.2.01$
IOS version is 17.2.1
TLV: T=9, L=53, V=CW_FULL_VERSION=$17.2.01.0.869.1580816579..Amsterdam$
TLV: T=9, L=52, V=CW_DESCRIPTION=$Cisco IOS Software, IOS-XE Software$
TLV: T=9, L=9, V=CW_END=$$
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.
```

Additional References for Boot Integrity Visibility

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9200 Series Switches)*

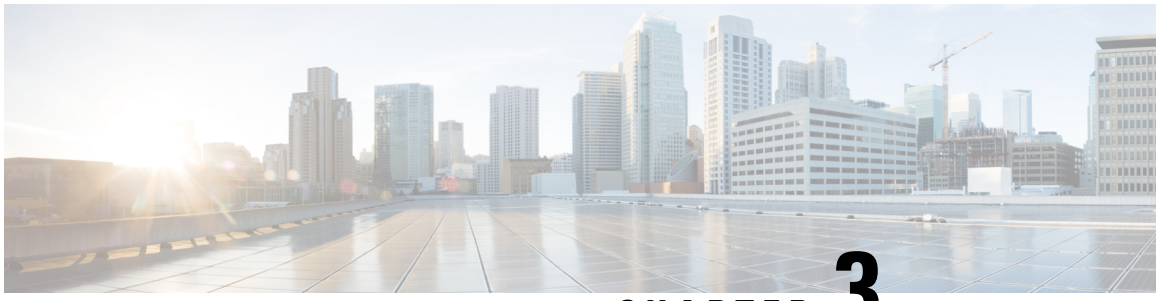
Feature History for Boot Integrity Visibility

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	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.
Cisco IOS XE Cupertino 17.9.1	Boot Integrity Visibility	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 3

Performing Device Setup Configuration

- Restrictions for Performing Device Setup Configuration, on page 49
- Information About Performing Device Setup Configuration, on page 49
- How to Perform Device Setup Configuration, on page 59
- Configuration Examples for Device Setup Configuration, on page 67
- Additional References For Performing Device Setup, on page 75
- Feature History for Performing Device Setup Configuration, on page 75

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 *Cisco Catalyst 9200 Series Switches 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.
- Stack interface loopback test to verify the stack-ring loopback functionality in the stacking environment.

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.

If the switch experiences 5 consecutive unexpected reloads within 15 minutes of startup, autoboot will be disabled, and the switch will enter ROMMON mode. To recover, issue the **boot** command manually. This prevents continuous boot loops and ensures system stability.

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 supports two modes to boot the software packages:

Installed Boot Mode

You can boot your 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 Catalyst 9200 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:  boot flash:cat9k_lite_iosxe.16.09.02.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:cat9k_2.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 49.

Default Switch Information

Table 4: 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.
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.

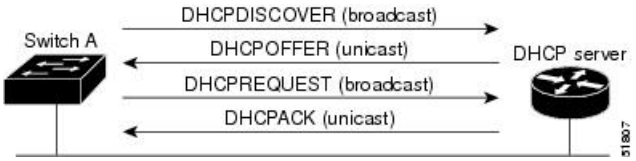
Note Certain device configurations and protocols are identified as insecure and pose a security risk to Cisco devices. While existing deployments continue to function, you must intentionally enable these features for new installations. For more information on remediation, refer to Cisco C9000 Switching IOS XE – Resilient Infrastructure Playbook.

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 2: 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.

Note Certain device configurations and protocols are identified as insecure and pose a security risk to Cisco devices. While existing deployments continue to function, you must intentionally enable these features for new installations. For more information on remediation, refer to Cisco C9000 Switching IOS XE – Resilient Infrastructure Playbook.

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`	Specifies the name of the configuration file that is used as a boot image.

	Command or Action	Purpose
	`config-boot.text`	
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 (/).
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.

	Command or Action	Purpose
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.

	Command or Action	Purpose
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** 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:** `Device> 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 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:**	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`Device# copy running-config startup-config`	

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:cat9k_lite_iosxe.16.09.01.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:**	Copies the software install package from a remote location (via FTP, HTTP, HTTPs,

	Command or Action	Purpose
	`Device# install add file tftp://172.16.0.1//tftpboot/folder1/cat9k_iosxe.16.06.01.SPA.bin`	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:** `Device# install activate`	Activates the added software install package, and reloads the device. • 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**	Displays information about the active package.

	Command or Action	Purpose
	Example: `Device# show install summary`	• 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:`
	`switch:`
	`switch:`
	`switch:BOOT=tftp://10.0.0.2/cat9k_lite_iosxe.16.09.02.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:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
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.

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 cat9k-rpboot.16.09.01.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 [Fuji], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.9.1, RELEASE
 SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2017 by Cisco Systems, Inc.
Compiled Tue 30-May-17 00:36 by mcpre


Cisco IOS-XE software, Copyright (c) 2005-2017 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 C9200L-24P-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                       : C9200L-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!
```

The following example displays software bootup in bundle mode:

```
switch: boot flash: cat9k_lite_iosxe.16.09.01.SPA.bin

Attempting to boot from [flash: cat9k_lite_iosxe.16.09.01.SPA.bin]
Located cat9k_lite_iosxe.16.09.01.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 [Fuji], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.9.1, RELEASE
```

```
 SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2017 by Cisco Systems, Inc.
Compiled Tue 30-May-17 00:36 by mcpre


Cisco IOS-XE software, Copyright (c) 2005-2017 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 C9200L-24P-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                       : C9200L-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:cat9k_lite_iosxe.16.09.01.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:cat9k_lite_iosxe.16.09.01.SPA.bininstall_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:cat9k_lite_iosxe.16.09.01.SPA.bin from switch 7 to switch 4
[4]: Finished copying to switch 4
Info: Finished copying flash:cat9k_lite_iosxe.16.09.01.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/cat9k_lite-webui.16.09.01.SPA.pkg
/flash/cat9k_lite-srdriver.16.09.01.SPA.pkg
/flash/cat9k_lite-rpboot.16.09.01.SPA.pkg
/flash/cat9k_lite-rpbase.16.09.01.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:cat9k_lite_iosxe.16.09.01.SPA.binAug 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
--------------------------------------------------------------------------------
```

The following example shows how to activate an added software package file:

The following sample output from the **show install summary** command displays the status of the software package as active and uncommitted:

The following example shows how to execute the **install commit** command:

The following example shows how to rollback an update package to the base package:

The following is a sample output from the **install remove inactive** command:

The following is sample output from the **install abort** command:

The following is a sample output from the **install activate auto-abort-timer** command:

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 Aug 30 06:39:48 Universal 2018
[0|install_op_boot]: END SUCCESS  Tue Aug 30 06:39:50 Universal 2018
```

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    16.9.1.0.70

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

Step 4 **show install package** *filesystem: filename*

Example:

```
Device# show install package flash:cat9k_lite-rpboot.16.09.01.SPA.pkg
```

Displays information about the specified software install package file.

```
Device# show install package flash:cat9k_lite-rpboot.16.09.01.SPA.pkg
Package: cat9k_lite-rpboot.16.09.01.SPA.pkg
  Size: 34616705
  Timestamp: Thu Aug 30 20:28:25 2018 UTC
  Canonical path: /flash/cat9k_lite-rpboot.16.09.01.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]
```

Additional References For Performing Device Setup

Related Documents

Related Topic	Document Title
Device setup commands Boot loader commands	*Command Reference (Catalyst 9200 Series Switches)*
Hardware installation	*Cisco Catalyst 9200 Series Switches Hardware Installation Guide* .

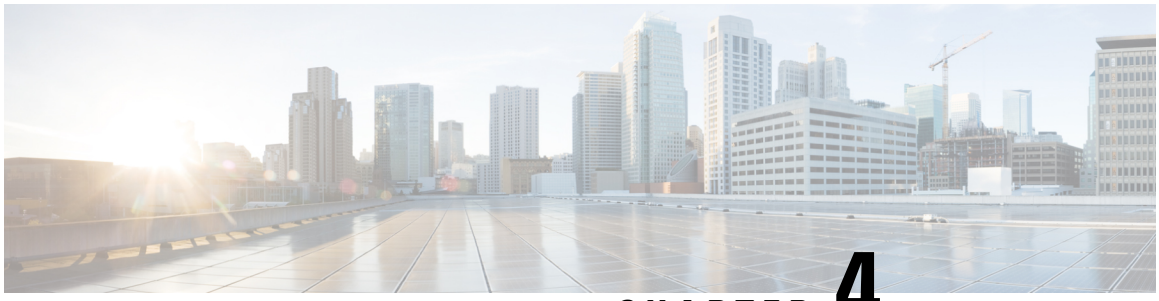
Feature History for Performing Device Setup Configuration

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	Device Setup Configuration	A device setup configuration can be performed, including auto configuration of IP address assignments and DHCP.
Cisco IOS XE Cupertino 17.9.1	Device Setup Configuration	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 4

Configuring Application Visibility and Control in a Wired Network

- Information About Application Visibility and Control in a Wired Network, on page 77
- Supported AVC Class Map and Policy Map Formats, on page 77
- Restrictions for Wired Application Visibility and Control, on page 79
- How to Configure Application Visibility and Control, on page 81
- Monitoring Application Visibility and Control, on page 106
- Examples: Application Visibility and Control Configuration, on page 106
- Basic Troubleshooting - Questions and Answers, on page 118
- Additional References for Application Visibility and Control, on page 119
- Feature History for Application Visibility and Control in a Wired Network, on page 120

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 as well as for a switch stack. 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 allowed only on wired physical ports. Policy configuration is not supported on virtual interfaces like VLAN and other logical interfaces.
- 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).
- AVC is not supported on management port (Gig 0/0).
- 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.
- Performance: Each switch member is able to handle 500 connections per second (CPS) at less than 50% CPU utilization.
- 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*).
- Starting with Cisco IOS XE 16.12.1 release, 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.
- For wired AVC traffic, four AVC flow monitors per direction, interface, and protocol (IPv4/6) are supported on the system.

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 1/0/1`	Specifies the interface for which you are enabling protocol-discovery and enters interface configuration mode.

	Command or Action	Purpose
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.

	Command or Action	Purpose
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.
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**

	Command or Action	Purpose
		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.
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 1/0/1`	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.

	Command or Action	Purpose
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:**	Specifies a match to the application name. **Note**

	Command or Action	Purpose
	`Device(config-flow-record)# match application name`	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:** `Device(config-flow-record)# match connection server transport port`	Specifies a match to the transport port of the server.
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**

	Command or Action	Purpose
		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:** `Device(config-flow-record)# collect connection client counter packets long`	Specifies to collect the number of packets sent by the client.
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.

	Command or Action	Purpose
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:**	Specifies a match to the IPv4 address of the client (flow initiator).

	Command or Action	Purpose
	`Device(config-flow-record)# match connection client ipv4 address`	
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).
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 :

	Command or Action	Purpose
		• 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.
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.

	Command or Action	Purpose
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:**	Specifies a match to the IPv4 destination address as a key field.

	Command or Action	Purpose
	`Device(config-flow-record)# match ipv4 destination address`	
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.
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.

	Command or Action	Purpose
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.

	Command or Action	Purpose
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.
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.

	Command or Action	Purpose
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:**	Specifies a match to the application name. **Note**

	Command or Action	Purpose
	`Device(config-flow-record)# match application name`	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:** `Device(config-flow-record)# match connection server ipv4 address`	Specifies a match to the IPv4 address of the server (flow responder).
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.

	Command or Action	Purpose
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.
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.

	Command or Action	Purpose
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.

	Command or Action	Purpose
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`	(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.

	Command or Action	Purpose

	Example:
	`Device(config-flow-monitor)# cache timeout inactive 200`
	Example:
	`Device(config-flow-monitor)# cache type normal`
		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:** `Device# show flow monitor`	Displays information about all the flow monitors.
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**	Displays flow cache contents in CSV format.

	Command or Action	Purpose
	Example: `Device# show flow monitor flow-monitor-1 cache format csv`	

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 1/0/1`	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.

Warning When using switch stacking, ensure that each switch has the same Protocol Pack file loaded. If you execute the **ip nbar protocol-pack flash** *protocol-pack-file* command on the primary switch in the stack, any switch in the stack that does not have the file loaded will be reloaded due to a configuration mismatch.

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 105 .

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:
	`Device(config)# default ip nbar protocol-pack`
		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.
		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 5: 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 1/0/1
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 GigabitEthernet1/0/2
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 GigabitEthernet1/0/1


GigabitEthernet1/0/1
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


GigabitEthernet1/0/1
 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 1/0/2
 GigabitEthernet1/0/2

  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 stack 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.

11. **Question:** I see a high and incrementing drop in the high-rate application queue when running **show controller cpu interface** or **show platform software fed switch active cpu-interface** commands. Is this a problem or traffic loss?

    Answer:

    This is an expected behavior when AVC is configured.

    When AVC is enabled, the first packet of each new flow triggers flow monitoring because NBAR is active on the interface. A NetFlow entry is created in the cache and marked with the "Copy to CPU" attribute. As part of this process, a duplicate of the first packet is punted to the CPU and then discarded, which is why drops are observed in the high-rate application queue. The original packet continues normal hardware forwarding, so there is no service interruption or traffic loss. When AVC is configured, this counter can be safely ignored, as this behavior is expected and does not indicate a fault or performance issue.

Additional References for Application Visibility and Control

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9200 Series Switches)*

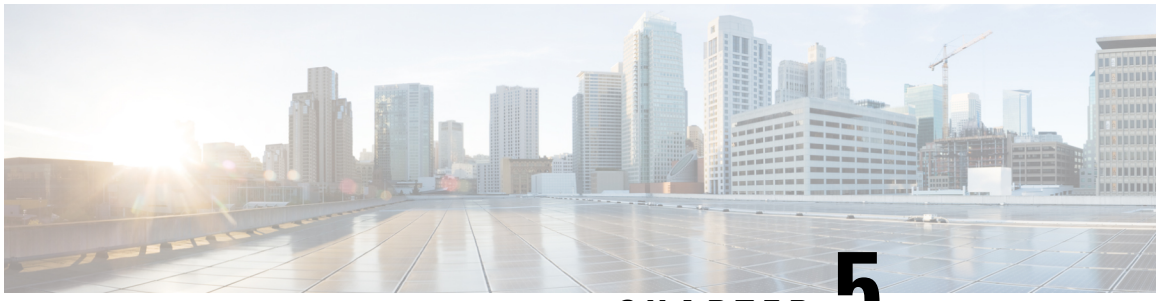
Feature History for Application Visibility and Control in a Wired Network

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.8.1a	Wired Application Visibility and Control (Wired AVC) Attribute-based QoS (EasyQoS)	Support for defining QoS classes and policies based on Network-Based Application Recognition (NBAR) attributes instead of specific protocols, was made available, with a few limitations. Only business-relevance and traffic-class are the supported NBAR attributes.
Cisco IOS XE Gibraltar 16.11.1	Application Visibility and Control in a Wired Network	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.
Cisco IOS XE Gibraltar 16.12.1	DNS flow record	Support for DNS flow record was introduced. DNS flow record uses the DNS Domain-Name as the collect field for defining the flow record.
Cisco IOS XE Amsterdam 17.3.1	Interoperability of Application Visibility and Control and Encrypted Traffic Analytics	Support for interoperability of Application Visibility and Control and Encrypted Traffic Analytics on the same port was introduced.
Cisco IOS XE Cupertino 17.9.1	Application Visibility and Control in a Wired Network	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 5

Configuring SDM Templates

- Information About SDM Templates, on page 121
- How to Configure SDM Templates, on page 121
- Monitoring and Maintaining SDM Templates, on page 122
- Configuration Examples for SDM Templates, on page 123
- Additional References for SDM Templates, on page 124
- Feature History for SDM Templates, on page 124

Information About SDM Templates

You can use SDM templates to configure system resources to optimize support for specific features, depending on how your device is used in the network. You can select a template to provide maximum system usage for some functions.

Cisco Catalyst 9200 Series Switches support the following templates:

- Advanced
- VLAN

It is recommended that you reload the system as soon as you make a change to the SDM template. After you change the template and the system reboots, you can use the **show sdm prefer** privileged EXEC command to verify the new template configuration. If you enter the **show sdm prefer** command before you enter the **reload** privileged EXEC command, the **show sdm prefer** command shows the template currently in use and the template that will become active after a reload.

How to Configure SDM Templates

Setting the SDM Template

Follow these steps to use the SDM template to maximize feature usage:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**sdm prefer** {**advanced** | **vlan**} **Example:** `Device(config)#sdm prefer vlan`	Selects an SDM template. • **advanced** —Sets the switch to the advanced template. • **vlan** —Maximizes VLAN configuration on the switch with no routing supported in hardware.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 5	**reload** **Example:** `Device# reload`	Reloads the operating system. After the system reboots, you can use the **show sdm prefer** privileged EXEC command to verify the new template configuration. If you enter the **show sdm prefer** command before you enter the reload privileged EXEC command, the **show sdm prefer** command shows the template currently in use and the template that will become active after a reload.

Monitoring and Maintaining SDM Templates

Verifying SDM Templates

Use the following commands to monitor and maintain SDM templates.

Command	Purpose
show sdm prefer	Displays the SDM template in use.

Note The SDM templates contain only those commands that are defined as part of the templates. If a template enables another related command that is not defined in the template, then this other command will be visible when the **show running config** command is entered. For example, if the SDM template enables the **switchport voice vlan** command, then the **spanning-tree portfast edge** command may also be enabled (although it is not defined on the SDM template).

If the SDM template is removed, then other such related commands are also removed and have to be reconfigured explicitly.

Configuration Examples for SDM Templates

Examples: Displaying SDM Templates

This is an example output showing the advanced template information.

```
Device# show sdm prefer advanced

Showing SDM Template Info

This is the Advanced template.
  Number of VLANs:                                4094
  Unicast MAC addresses:                          16384
  Overflow Unicast MAC addresses:                 256
  L2 Multicast entries:                           1024
  L3 Multicast entries:                           1024
  Overflow L3 Multicast entries:                  256
  Directly connected routes:                      10240
  Indirect routes:                                4096
  Security Access Control Entries:                1664
  QoS Access Control Entries:                     1024
  Policy Based Routing ACEs:                      512
  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.
```

This is an example output showing the VLAN template information.

```
Device# show sdm prefer vlan

Showing SDM Template Info

This is the VLAN template for a typical Layer 2 network.
  Number of VLANs:                                  4094
  Unicast MAC addresses:                            32768
  Overflow Unicast MAC addresses:                   256
  L2 Multicast entries:                             1024
  L3 Multicast entries:                             1024
  Overflow L3 Multicast entries:                    256
  Direct/Indirect shared unicast routes:            6144
  Security Access Control Entries:                  1664
  QoS Access Control Entries:                       1024
  Policy Based Routing ACEs:                        512
  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.
```

Additional References for SDM Templates

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9200 Series Switches)*

Feature History for SDM Templates

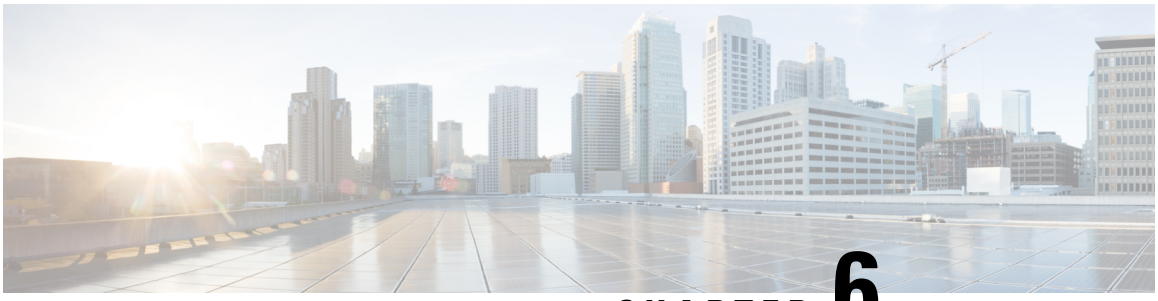
This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	SDM Template	Standard SDM templates can be used to configure system resources to optimize support for specific features.

Release	Feature	Feature Information
Cisco IOS XE Cupertino 17.9.1	SDM Template	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to Cisco Feature Navigator.



CHAPTER 6

Configuring System Message Logs

- Information About Configuring System Message Logs, on page 127
- How to Configure System Message Logs, on page 130
- Monitoring and Maintaining System Message Logs, on page 137
- Configuration Examples for System Message Logs, on page 137
- Additional References for System Message Logs, on page 138
- Feature History for System Message Logs, on page 138

Information About Configuring System Message Logs

System Messsage Logging

Note Certain device configurations and protocols are identified as insecure and pose a security risk to Cisco devices. While existing deployments continue to function, you must intentionally enable these features for new installations. For more information on remediation, refer to Cisco C9000 Switching IOSXE – Resilient Infrastructure Playbook.

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 6: 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. **Note** Certain device configurations and protocols are identified as insecure and pose a security risk to Cisco devices. While existing deployments continue to function, you must intentionally enable these features for new installations. For more information on remediation, refer to Cisco C9000 Switching IOS XE – Resilient Infrastructure Playbook.
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 7: Default System Message Logging Settings

Feature	Default Setting
System message logging to the console	Enabled.
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.

	Command or Action	Purpose
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

	Command or Action	Purpose
		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.
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:** `Device# configure terminal`	Enters global configuration mode.
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`	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.

	Command or Action	Purpose
	or `Device(config)# service timestamps log datetime`	**Note** Certain device configurations and protocols are identified as insecure and pose a security risk to Cisco devices. While existing deployments continue to function, you must intentionally enable these features for new installations. For more information on remediation, refer to Cisco C9000 Switching IOS XE – Resilient Infrastructure Playbook.
Step 3	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

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:** `Device# configure terminal`	Enters global configuration mode.
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.
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.

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: Stacking System Message

This example shows a partial switch system message for an active switch and a stack member (hostname *Switch-2*):

```
00:00:46: %LINK-3-UPDOWN: Interface Port-channel1, changed state to up
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/1, changed state to up
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/2, 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 GigabitEthernet1/0/1, 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)

00:00:46: %LINK-3-UPDOWN: Interface Port-channel1, changed state to up (Switch-2)
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet2/0/1, changed state to up (Switch-2)
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet2/0/2, changed state to up (Switch-2)
00:00:48: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state to down
(Switch-2)
00:00:48: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet2/0/1, changed
state to down 2 (Switch-2)
```

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 GigabitEthernet0/1, changed state to up
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet0/2, 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 GigabitEthernet0/1, 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)
```

Additional References for System Message Logs

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9200 Series Switches)*

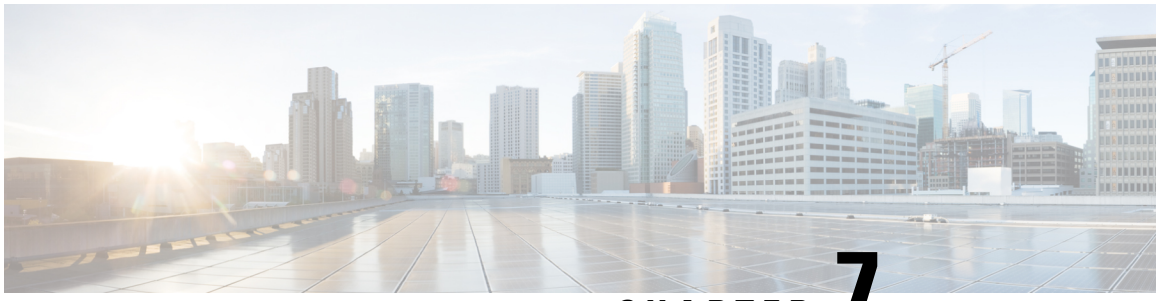
Feature History for System Message Logs

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	System Message Logs	A switch sends the output from system messages 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
Cisco IOS XE Cupertino 17.9.1	System Message Logs	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 7

Configuring Online Diagnostics

- Restrictions for Online Diagnostics, on page 141
- Information About Configuring Online Diagnostics, on page 141
- How to Configure Online Diagnostics, on page 145
- Monitoring and Maintaining Online Diagnostics, on page 149
- Configuration Examples for Online Diagnostics, on page 150
- Additional References for Online Diagnostics, on page 152
- Feature History for Configuring Online Diagnostics, on page 152

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.
Intitial release	Cisco IOS XE Fuji 16.9.2.
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.
Intitial release	Cisco IOS XE Fuji 16.9.2.
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.
Intitial release	Cisco IOS XE Fuji 16.9.2.
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.
Intitial release	Cisco IOS XE Fuji 16.9.2.
Corrective action	–
Hardware support	All modules.

DiagPoETest

This test checks the Power over Ethernet (PoE) controller functionality. Do not perform this test during normal switch operation.

Attribute	Description
Disruptive or Nondisruptive	Disruptive.

Attribute	Description
Recommendation	Run this test if you experience PoE controller issues with a port. This can be run only as an on-demand test.
Default	Off.
Intitial release	Cisco IOS XE Fuji 16.9.2.
Corrective action	–
Hardware support	All modules.

DiagStackCableTest

This test verifies the stack-ring loopback functionality in the stacking environment. It cannot be run as a health-monitoring test.

Attribute	Description
Disruptive or Nondisruptive	Disruptive.
Recommendation	Run this test to verify the stack-ring loopback functionality in the stacking environment.
Default	Off.
Intitial release	Cisco IOS XE Fuji 16.9.2.
Corrective action	If the test fails, check the stack cables and connectors.
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.
Intitial release	Cisco IOS XE Fuji 16.9.2.
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 or a device stack and check the test results by using the privileged EXEC **show** commands in this table:

Table 8: 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.

Examples: Start Diagnostic Tests

This example shows how to start a diagnostic test by using the test name:

```
Device# diagnostic start switch 2 test DiagPOETest
```

This example shows how to start all of the basic diagnostic tests:

```
Device# diagnostic start switch 1 test all
```

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.

DiagFanTest :
        This test verifies all fan modules have been inserted and working properly on the
board
        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#
```

Additional References for Online Diagnostics

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9200 Series Switches)*

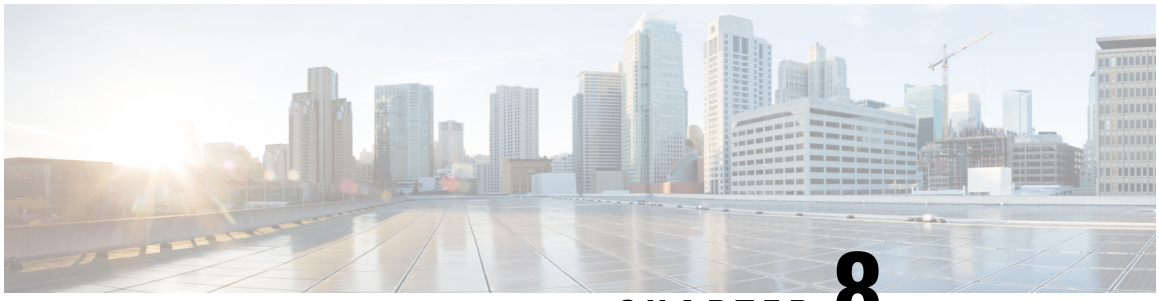
Feature History for Configuring Online Diagnostics

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	Online Diagnostics	With online diagnostics, you can test and verify the hardware functionality of the device while the device is connected to a live network.
Cisco IOS XE Cupertino 17.9.1	Online Diagnostics	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 8

Consistency Checker

- Limitations for Consistency Checker, on page 153
- Information about Consistency Checker, on page 154
- Running the Consistency Checker, on page 155
- Output Examples for Consistency Checker, on page 155
- Feature History for Consistency Checker, on page 161

Limitations for Consistency Checker

The Consistency Checker has the following limitations:

- Consistency Checkers are CPU intensive. It is not recommended to run the checkers at very short intervals.
- Legacy Consistency Checkers do not have support for snapshot. So, the previous runs cannot be displayed.
- There is no command to stop/abort the already running Consistency Checkers.
- Forwarding Engine hardware entry validations are partially implemented. Only programming failures can be detected and reported.
- Layer2 MAC Consistency Checker can validate the MAC address in hardware with software copy.
- Consistency checker is designed to reduce false positives in all cases. However, there could be rare cases of reporting a false positive in the following scenarios:
  - Large table state changes (i.e clear, relearn etc).
  - Under very high CPU usage due to any other feature while a consistency checker running. The consistency checker may report inconsistency in processes where CPU usage is high.
- Forwarding engine hardware (FED) check is not entirely supported in Layer3 Multicast Consistency Checker. You can only detect and report on programming failures.
- Forwarding Manager-RP software entry is not supported in Layer3 Multicast Consistency Checker.

Information about Consistency Checker

Overview of Consistency Checker

The Consistency Checker collects information on various table states within the software and the hardware. It compares the software state with the hardware state. If there is any inconsistency, it flags the issue immediately. This helps to reduce increased troubleshooting time at a later period. The consistency checker supplements basic troubleshooting and helps to identify scenarios where inconsistent states between software and hardware tables are causing issues in the network, thereby reducing the mean time to resolve the issue.

There are two types of consistency checker implementation available:

- Legacy Consistency Checker - supports validating the entry from control plane to the forwarding engine (or hardware copy).
- End-to-End Consistency Checker - supports validating the software entry from control plane to all processes involved in distributing and handling the entry, as well as the forwarding engine's hardware copy.

End-to-End Consistency Checker

End-to-End (E2E) Consistency Checker supports full scan and single entry and should be started manually or run via gold diagnostic. The consistency checker can be started for a single entry using the command which helps to isolate the issue at which forwarding process entry is not consistent and helps speed up the debugging.

Every time the consistency checker is started, a runID is provided. Using the runID, its status, summary, details can be viewed. The last 5 snapshots are available any time for you to check the previous run's result.

E2E consistency checker performs the following functions:

- Validates the IOS entry to software tables/processes (Forwarding manger-RP, Forwarding manager-FP and FED) for all modules.
- Reports various inconsistencies (entry inconsistent, entry missing, stale entry) and sends a syslog to alert the administrator.
- Helps to speed up the fault isolation.
- Records any inconsistent entry with relevant data.
- Consistency checker supports the recursive single entry check which can validate the dependent objects along with the actual entry. (i.e, A Layer 3 Multicast with N outgoing interfaces can be validated for multicast entries along with OIFs programming, OIF's Adjacency validation, etc)
- Constant memory usages irrespective of total entries in a table.

Note The consistency checker is bound to CPU utilization and can not exceed the configured value while validating the tables across processes.

Features Supported in Consistency Checker

The following features are supported in consistency checker:

- Legacy Consistency Checker
  - **Layer2 MAC Consistency Checker**: This consistency checker validates the IOS entry to FED software entry. It also validates the MAC address into hardware tables.
  - **Layer3 FMANFP Entry Consistency Checker**: This consistency checker validates the Layer 2, Layer 3, and multicast objects status in the Forwarding Manager-FP process. This includes stale objects and long pending objects.
- E2E Consistency Checker
  - **Layer2 Multicast Consistency Checker**: This consistency checker validates the IOS Layer 2 multicast IGMP/MLD VLAN, the group entry to Forwarding Manager-FP software entry, FED software entry, and FED hardware programming errors.
  - **Layer3 Multicast Consistency Checker**: This consistency checker validates the IOS Layer 3 multicast IGMP/MLD VLAN, the group entry to Forwarding Manager-FP software entry and FED software entry.

Running the Consistency Checker

The table shown below lists the commands to run the various consistency checkers:

Command	Purpose
show consistency-checker l2	Runs the consistency-checker on the Layer 2 forwarding tables.
show consistency-checker l3	Runs the consistency-checker on the Layer 3 forwarding tables.
show consistency-checker mcast l2m	Runs the consistency-checker on the Layer 2 multicast forwarding tables.
show consistency-checker mcast l3m	Runs the consistency-checker on the Layer 3 multicast forwarding tables.
show consistency-checker objects	Runs the End-to-End consistency-checker on objects.
show consistency-checker run-id *run-id*	Runs the End-to-End consistency-checker by run ID.
show consistency-checker switch	Runs the consistency-checker on the specified switch.

Output Examples for Consistency Checker

The following is a sample output for the **show consistency-checker mcast l2m** command where the consistency checker runs a full scan:

```
Device# show consistency-checker mcast l2m start all
L2 multicast Full scan started. Run_id: 2
Use 'show consistency-checker run-id 2 status' for completion status.

Device#
*Feb 17 06:19:14.889: %FED_CCK_ERRMSG-4-INCONSISTENCY_FOUND: F0/0: fed: Consistency
Checker(CCK) detected inconsistency for l2m_vlan. Check 'show consistency run-id 2 detail'.
*Feb 17 06:19:14.890: %FED_CCK_ERRMSG-4-INCONSISTENCY_FOUND: F0/0: fed: Consistency
Checker(CCK) detected inconsistency for l2m_group. Check 'show consistency run-id 2 detail'.
Device#
*Feb 17 06:19:19.432: %IOSXE_FMANRP_CCK-6-FMANRP_COMPLETED: Consistency Check for Run-Id 2
 is completed. Check 'show consistency-checker run-id 2'.
Device#
Device# show consistency-checker run-id 2 status
Process: IOSD
  Object-Type      Status           Time(sec)     Exceptions
  l2m_vlan         Completed        13            No
  l2m_group        Completed        13            No

Process: FMAN-FP
  Object-Type      Status           Time(sec)     State
  l2m_vlan         Completed        9             Consistent
  l2m_group        Completed        9             Consistent

Process: FED
  Object-Type      Status           Time(sec)     State
  l2m_vlan         Completed        9             Inconsistent
  l2m_group        Completed        9             Inconsistent

Device#
Device# show consistency-checker run-id 2
Process: IOSD
  Object-Type    Start-time              Entries       Exceptions
  l2m_vlan       2021/02/17 06:19:05          22                0
  l2m_group      2021/02/17 06:19:05          24                0

Process: FMAN-FP
  *Statistics(A/I/M/S/Oth): Actual/Inherited/Missing/Stale/Others

  Object-Type    Start-time              State             A/  I/  M/  S/Oth
  l2m_vlan       2021/02/17 06:19:05     Consistent        0/  0/  0/  0/  0
  l2m_group      2021/02/17 06:19:05     Consistent        0/  0/  0/  0/  0

Process: FED
  *Statistics(A/I/M/S/HW/Oth): Actual/Inherited/Missing/Stale/Hardware/Others

  Object-Type    Start-time              State              A/  I/  M/  S/ HW/Oth
  l2m_vlan       2021/02/17 06:19:05     Inconsistent       1/  0/  0/168/  0/  0
  l2m_group      2021/02/17 06:19:05     Inconsistent       4/  0/  2/  0/  0/  0


Device#
Device# show consistency-checker run-id 2 detail
Process: IOSD

Process: FMAN-FP

Process: FED
  Object-Type:l2m_vlan   Start-time:2021/02/17 06:19:05
    Status:Completed   State:Inconsistent
    Key/data                                Reason
    (Ipv4, vlan: 768)                       Stale
      snoop:off stp_tcn:off flood:off pimsn:off
    (Ipv4, vlan: 769)                       Stale
```

```
      snoop:off stp_tcn:off flood:off pimsn:off
    (Ipv6, vlan: 900)                       Inconsistent
      snoop:on stp_tcn:on flood:on pimsn:off
    (Ipv6, vlan: 767)                       Stale
      snoop:off stp_tcn:off flood:off pimsn:off

  Object-Type:l2m_group   Start-time:2021/02/17 06:19:05
    Status:Completed   State:Inconsistent
    Key/data                                Reason
    (Ipv4, vlan:100 (*,227.0.0.0))          Inconsistent
      Group ports: total entries: 0
    (Ipv4, vlan:100 (*,227.1.0.0))          Missing

Device#
```

The following is a sample output for the **show consistency-checker mcast l2m** command where the consistency checker runs a recursive single-entry scan:

```
Device# show consistency-checker mcast l2m start vlan 900 229.1.1.1 recursive
Single entry scan started with Run_id: 2

*Feb 17 06:54:09.880: %IOSXE_FMANRP_CCK-6-FMANRP_COMPLETED: Consistency Check for Run-Id 2
 is completed.
Check 'show consistency-checker run-id 2'.
Device#
Device# show consistency-checker run-id 2
Process: IOSD
  Object-Type    Start-time              Entries       Exceptions
  l2m_vlan       2021/02/17 06:54:01           1                0
  l2m_group      2021/02/17 06:54:01           1                0

Process: FMAN-FP
  *Statistics(A/I/M/S/O): Actual/Inherited/Missing/Stale/Others

  Object-Type    Start-time              State            A / I / M / S / O
  l2m_vlan       1970/01/01 00:10:03     Consistent        0/  0/  0/  0/  0
  l2m_group      1970/01/01 00:10:03     Consistent        0/  0/  0/  0/  0


Process: FED
  *Statistics(A/I/M/S/HW/O): Actual/Inherited/Missing/Stale/Hardware/Others

  Object-Type    Start-time              State             A / I / M / S / HW/ O
  l2m_vlan       2021/02/17 06:54:01     Inconsistent       1/  0/  0/  0/  0/  0
  l2m_group      2021/02/17 06:54:01     Inconsistent       0/  1/  0/  0/  0/  0

Device#
Device# show consistency-checker run-id 2 detail
Process: IOSD
  Object-Type:l2m_vlan   Start-time:2021/02/17 06:54:01
    Key/data                                Reason
    (Ipv4, vlan:900)                        Success
      snoop:on stp_tcn:off flood:off pimsn:off

  Object-Type:l2m_group   Start-time:2021/02/17 06:54:01
    Key/data                                Reason
    (Ipv4, vlan:900, (*,229.1.1.1))         Success
      Twe1/0/5


Process: FMAN-FP

Process: FED
  Object-Type:l2m_group   Start-time:2021/02/17 06:54:01
```

```
    Status:Completed   State:Inconsistent
    Key/data                                Reason
    (Ipv4, vlan:900 (*,229.1.1.1))          Inherited
      Group ports: total entries: 1
        TwentyFiveGigE1/0/5

      ---------------Recursion-level-1-----------------
      Object-Type:l2m_vlan   Start-time:2021/02/17 06:54:01
        Status:Completed   State:Inconsistent
        Key/data                                Reason
        (Ipv4, vlan: 900)                       Inconsistent
          snoop:on stp_tcn:off flood:on pimsn:off

Device#
```

The following is a sample output for the **show consistency-checker objects** command where the consistency checker runs a scan on objects:

```
Device# show consistency-checker objects l2m_group
Process: IOSD
  Run-id    Start-time              Exception
  1         2021/02/17 05:20:42     0
  2         2021/02/17 06:19:05     0

Process: FMAN-FP
  *Statistics(A/I/M/S/Oth): Actual/Inherited/Missing/Stale/Others

  Run-id    Start-time              State           A/  I/  M/  S/Oth
  1         2021/02/17 05:20:42     Consistent      0/  0/  0/  0/  0
  2         2021/02/17 06:19:05     Consistent      0/  0/  0/  0/  0

Process: FED
  *Statistics(A/I/M/S/HW/Oth): Actual/Inherited/Missing/Stale/Hardware/Others

  Run-id    Start-time              State           A/  I/  M/  S/ HW/Oth
  1         2021/02/17 05:20:42     Consistent      0/  0/  0/  0/  0/  0
  2         2021/02/17 06:19:05     Inconsistent    4/  0/  2/  0/  0/  0

Device#
Stark#sh consistency-checker run 2 detail
Process: IOSD
  Object-Type:l2m_vlan   Start-time:2021/02/17 06:54:01
    Key/data                                Reason
    (Ipv4, vlan:900)                        Success
      snoop:on stp_tcn:off flood:off pimsn:off

  Object-Type:l2m_group   Start-time:2021/02/17 06:54:01
    Key/data                                Reason
    (Ipv4, vlan:900, (*,229.1.1.1))         Success
      Twe1/0/5


Process: FMAN-FP

Process: FED
  Object-Type:l2m_group   Start-time:2021/02/17 06:54:01
    Status:Completed   State:Inconsistent
    Key/data                                Reason
    (Ipv4, vlan:900 (*,229.1.1.1))          Inherited
      Group ports: total entries: 1
        TwentyFiveGigE1/0/5

      ---------------Recursion-level-1-----------------
      Object-Type:l2m_vlan   Start-time:2021/02/17 06:54:01
```

```
        Status:Completed   State:Inconsistent
        Key/data                                Reason
        (Ipv4, vlan: 900)                       Inconsistent
          snoop:on stp_tcn:off flood:on pimsn:off

Device# show consistency-checker objects l2m_group 2 detail
Process: IOSD

Process: FMAN-FP

Process: FED
  Object-Type:l2m_group   Start-time:2021/02/17 06:19:05
    Status:Completed   State:Inconsistent
    Key/data                                Reason
    (Ipv4, vlan:100 (*,227.0.0.0))          Inconsistent
      Group ports: total entries: 0
    (Ipv4, vlan:100 (*,227.1.0.0))          Missing
    (Ipv4, vlan:100 (*,227.0.0.1))          Inconsistent
      Group ports: total entries: 0
    (Ipv4, vlan:100 (*,227.1.0.1))          Missing
    (Ipv4, vlan:100 (*,227.0.0.2))          Inconsistent
      Group ports: total entries: 0
    (Ipv4, vlan:100 (*,227.0.0.3))          Inconsistent
      Group ports: total entries: 0


Device#
```

The following is a sample output for the **show consistency-checker mcast l3m** command where the consistency checker runs a full scan:

```
Device#sh consistency-checker mcast l3m start all
L3 multicast Full scan started. Run_id: 1
Use 'show consistency-checker run-id 1 status' for completion status.

Device#
*Apr  2 17:30:01.831: %IOSXE_FMANRP_CCK-6-FMANRP_COMPLETED: Consistency Check for Run-Id 1
 is completed. Check 'show consistency-checker run-id 1'.
Device#sh consistency-checker run-id 1
Process: IOSD
Flags:    F - Full Table Scan, S - Single Entry Run
          RE -  Recursive Check, GD -  Garbage Detector
          Hw -  Hardware Check, HS - Hardware Shadow Copy
  Object-Type    Start-time              Entries  Exceptions  Flags
  l3m_entry      2021/04/02 17:29:35           8         0    F GD Hw HS

Process: FMAN-FP
  *Statistics(A/I/M/S/Oth): Actual/Inherited/Missing/Stale/Others

  Object-Type    Start-time              State             A/  I/  M/  S/Oth
  l3m_entry      2021/04/02 17:29:35     Consistent        0/  0/  0/  0/  0

Process: FED
  *Statistics(A/I/M/S/HW/Oth): Actual/Inherited/Missing/Stale/Hardware/Others

  Object-Type    Start-time              State              A/  I/  M/  S/ HW/Oth
  l3m_entry      2021/04/02 17:29:35     Consistent         0/  0/  0/  0/  0/  0

Device#sh consistency-checker mcast l3m start 225.1.1.1 recursive
Single entry scan started with Run_id: 2
Use 'show consistency-checker run-id 2 status' for completion status.

Device#sh consistency-checker run-id 2 status
Process: IOSD
```

```
  Object-Type      Status           Time(sec)     Exceptions
  l2m_vlan         Completed        11            No
  l2m_group        Completed        11            No
  l3m_entry        Completed        11            No

Process: FMAN-FP
  Object-Type      Status           Time(sec)     State
  l2m_vlan         Completed        12            Consistent
  l2m_group        Completed        12            Consistent
  l3m_entry        Completed        12            Consistent

Process: FED
  Object-Type      Status           Time(sec)     State
  l2m_vlan         Completed        12            Consistent
  l2m_group        Completed        12            Consistent
  l3m_entry        Completed        12            Consistent

Device#sh consistency-checker run-id 2 detail
Process: IOSD
  Object-Type:l2m_vlan   Start-time:2021/04/02 17:34:12
    Key/data                                 Reason
    (Ipv4, vlan:100)                         Success
      snoop:on stp_tcn:off flood:off pimsn:off

  Object-Type:l2m_group   Start-time:2021/04/02 17:34:12
    Key/data                                 Reason
    (Ipv4, vlan:100, (*,225.1.1.1))          Success
      Fo1/0/3

  Object-Type:l3m_entry   Start-time:2021/04/02 17:34:12
    Key/data                                 Reason
    (Ipv4, (*,225.1.1.1))                    Success
      Entry flags: C
      Total entries: 1
      Obj_id: F80004A1 Obj_flags: F

Process: FMAN-FP
Process: FED
```

The following is a sample output for the **show consistency-checker mcast l3m** command where the consistency checker runs a recursive single-entry scan:

```
Device#sh consistency-checker mcast l3m start 225.1.1.1 15.1.1.1 recursive
Single entry scan started with Run_id: 4
Use 'show consistency-checker run-id 4 status' for completion status.
Device#sh consistency-checker run-id 4 status
Process: IOSD
  Object-Type      Status           Time(sec)     Exceptions
  l2m_vlan         Completed        10            No
  l2m_group        Completed        10            No
  l3m_entry        Completed        10            No

Process: FMAN-FP
  Object-Type      Status           Time(sec)     State
  l2m_vlan         Completed        11            Consistent
  l2m_group        Completed        11            Consistent
  l3m_entry        Completed        11            Consistent

Process: FED
  Object-Type      Status           Time(sec)     State
  l2m_vlan         Completed        11            Consistent
  l2m_group        Completed        11            Consistent
  l3m_entry        Completed        11            Consistent
Device#sh consistency-checker run-id 4 detail
Process: IOSD
```

```
  Object-Type:l2m_vlan   Start-time:2021/04/02 17:37:36
    Key/data                                Reason
    (Ipv4, vlan:100)                        Success
      snoop:on stp_tcn:off flood:off pimsn:off

  Object-Type:l2m_group   Start-time:2021/04/02 17:37:36
    Key/data                                Reason
    (Ipv4, vlan:100, (*,225.1.1.1))         Success
      Fo1/0/3

  Object-Type:l3m_entry   Start-time:2021/04/02 17:37:36
    Key/data                                Reason
    (Ipv4, vrf:, (15.1.1.1,225.1.1.1))      Success
      Entry flags:
      Total entries: 2
      Obj_id: F80004A1 Obj_flags: F
      Obj_id: F80003C1 Obj_flags: A

Process: FMAN-FP
Process: FED
```

The following is a sample output for the **show diagnostic content** command where end to end consistency is checked through gold diagnostics:

```
Device#show diagnostic content switch all

switch 2  module 1:

  Diagnostics test suite attributes:
    M/C/* - Minimal bootup level test / Complete bootup level test / NA
      B/* - Basic ondemand test / NA
    P/V/* - Per port test / Per device test / NA
    D/N/* - Disruptive test / Non-disruptive test / NA
      S/* - Only applicable to standby unit / NA
      X/* - Not a health monitoring test / NA
      F/* - Fixed monitoring interval test / NA
      E/* - Always enabled monitoring test / NA
      A/I - Monitoring is active / Monitoring is inactive

                                                          Test Interval   Thre-
  ID   Test Name                           Attributes     day hh:mm:ss.ms shold
  ==== ================================== ============   =============== =====
    1) TestGoldPktLoopback -------------> *BPN*X**I      not configured  n/a
    2) TestOBFL ------------------------> *B*N*X**I      not configured  n/a
    3) TestFantray ---------------------> *B*N****A      000 00:01:40.00 1
    4) TestPhyLoopback -----------------> *BPD*X**I      not configured  n/a
    5) TestThermal ---------------------> *B*N****A      000 00:01:30.00 1
    6) TestScratchRegister -------------> *B*N****A      000 00:01:30.00 5
    7) TestPortTxMonitoring ------------> *BPN****A      000 00:02:30.00 1
    8) TestConsistencyCheckL2 ----------> *B*N****A      000 00:01:30.00 1
    9) TestConsistencyCheckL3 ----------> *B*N****A      000 00:01:30.00 1
   10) TestConsistencyCheckMcast -------> *B*N****A      000 00:01:30.00 1
   11) TestConsistencyCheckL2m ---------> *B*N****A      000 00:01:30.00 1
   12) TestConsistencyCheckL3m ---------> *B*N****A      000 00:01:30.00 1
This gives the status of consistency check for multicast
```

Feature History for Consistency Checker

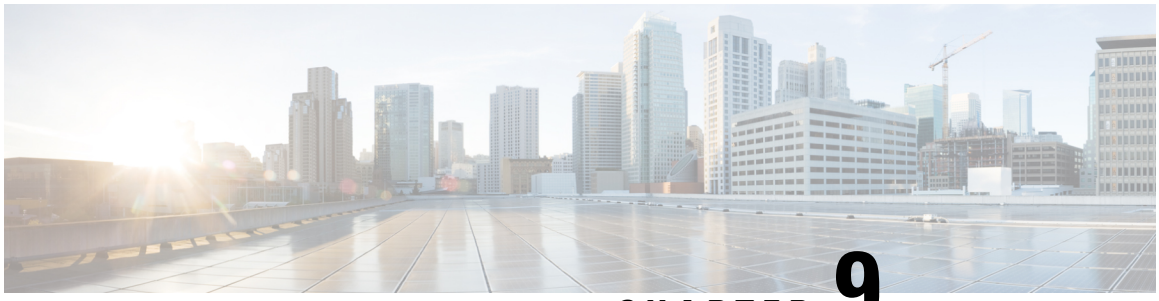
This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Amsterdam 17.3.1	Consistency Checker	The Consistency Checker collects information on various table states within the software and the hardware and flags any inconsistency it finds immediately. It supplements basic troubleshooting and helps to identify scenarios where inconsistent states between software and hardware tables are causing issues in the network, thereby reducing the mean time to resolve the issue.
Cisco IOS XE Bengaluru 17.6.1	Consistency Checker	This feature was enhanced and the multicast consistency checkers were introduced. The following keywords were added to the **show consistency-checker** command: **mcast**, **objects**, and **run-id**.
Cisco IOS XE Cupertino 17.9.1	Consistency Checker	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/

http://www.cisco.com/go/cfn.



CHAPTER 9

Managing Configuration Files

- Prerequisites for Managing Configuration Files, on page 163
- Restrictions for Managing Configuration Files, on page 163
- Information About Managing Configuration Files, on page 163
- How to Manage Configuration File Information, on page 170
- Feature History for Managing Configuration Files, on page 197

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 171 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 192 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.

The **service compress-config** global configuration command works only if you have Cisco IOS software Release 10.0 or later release boot ROMs. Installing new ROMs is a one-time operation and is necessary only if you do not already have Cisco IOS Release 10.0 in ROM. If the boot ROMs do not recognize a compressed configuration, the following message is displayed:

```
Boot ROMs do not support NVRAM compression Config NOT written to NVRAM
```

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 192 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 172 and Configuring the Device to Download Configuration Files, on page 169 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.)

	Command or Action	Purpose
		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 the necessary configuration commands. The Cisco IOS documentation set describes configuration commands organized by technology.
Step 4	Do one of the following: • **end** • **^Z**	Ends the configuration session and exits to EXEC mode. **Note**

	Command or Action	Purpose
	Example: `Device(config)# end`	When you press the Ctrl and Z keys simultaneously, ^Z is displayed to the screen.
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 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.
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**	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 rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username NetAdmin1`	(Optional) Changes the default remote username.
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.

	Command or Action	Purpose
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:** `Device# copy system:running-config ftp:`	Copies the running configuration or startup configuration file to the specified location on the FTP server.

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** **Example:** `Device> enable`	Enables privileged EXEC mode. • 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`	Copies the configuration file from an rcp server to the running configuration or startup configuration.

	Command or Action	Purpose
	`rcp://[user1@example.com/dir10/fileone] nvram:startup-config`	

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**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• 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:** `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 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.

	Command or Action	Purpose
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]."
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.
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.

	Command or Action	Purpose
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.

	Command or Action	Purpose
Step 4	**boot system** {**ftp:**[[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] | **rcp:**[[[//[*username***@**]*location*]/*directory*]/*filename*] | **tftp:**[[[//*location*]/*directory*]/*filename*]} **Example:** `Device(config)# boot system 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** **Example:** `Device# copy system:running-config nvram:startup-config`	Saves the configuration.

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:	• Loads a configuration file directly into NVRAM or

	Command or Action	Purpose
	• **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`	• 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*]	Copies a configuration file between flash memory devices.

	Command or Action	Purpose
	Example: `Device# copy flash: usbflash0:`	• 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... eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee ...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**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• 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).
Step 6	**copy ftp:** [[*//location*]*/directory*]*/bundle_name* **flash:** **Example:** `Device>copy ftp:/cat9k_iosxe.16.11.01.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**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• 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 rcp://netadmin@172.16.101.101/bundle1 flash:`	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 **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`	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.

	Command or Action	Purpose
	tftp://cat3k_caa-universalk9.SSA.03.12.02.EZP.150-12.02.EZP.150-12.02.EZP.bin flash:	

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.
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.

	Command or Action	Purpose
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

	Command or Action	Purpose
		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:** `Device# copy system:running-config nvram:startup-config`	Copies the configuration file to the flash file system from which the device loads the file on restart.
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:**	Saves the configuration performed in Step 3 to the startup configuration.

	Command or Action	Purpose
	`Device# copy system:running-config nvram:startup-config`	
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 Device to Download the Network Configuration File , on page 194
- Configuring the Device to Download the Host Configuration File , on page 195

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 system** {**ftp:**[[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*] | **rcp:**[[[//[*username***@**]*location*]/*directory*]/*filename*] | **tftp:**[[[//*location*]/*directory*]/*filename*]} **Example:** `Device(config)# boot system 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

	Command or Action	Purpose
		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:**	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

	Command or Action	Purpose
	`Device(config)# boot host tftp:hostfile1`	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.
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 system tftp:networkfile1
Device(config)# service config
Device(config)# end
Device# copy system:running-config nvram:startup-config
```

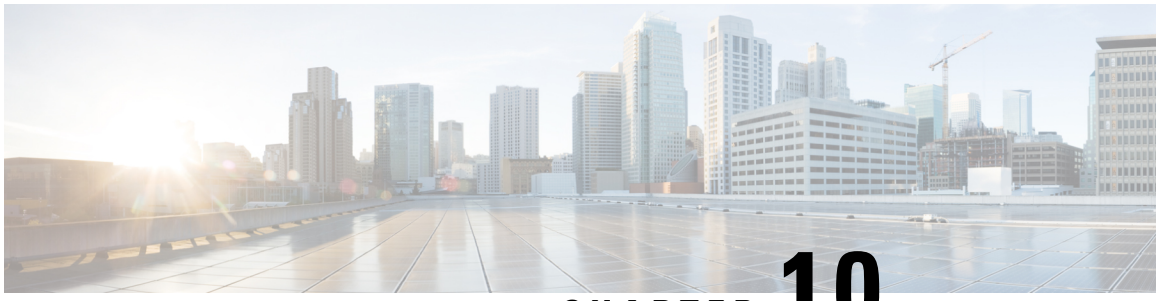
Feature History for Managing Configuration Files

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	Managing 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.
Cisco IOS XE Cupertino 17.9.1	Managing Configuration Files	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 10

Secure Copy

This document provides the procedure to configure a Cisco device for Secure Copy (SCP) server-side functionality.

- Prerequisites for Secure Copy, on page 199
- Information About Secure Copy, on page 199
- How to Configure Secure Copy, on page 200
- Configuration Examples for Secure Copy, on page 204
- Additional References for Secure Copy, on page 204
- Feature History for Secure Copy, on page 205

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. Beginning from Cisco IOS XE Dublin 17.10.1, 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:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Sets AAA authentication at login.
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:**	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 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:**	Sets AAA authentication to use the local username database for authentication at login.

	Command or Action	Purpose
	`Device(config)# aaa authentication login default local`	
Step 5	**aaa authorization exec default local** **Example:** `Device(config)# aaa authorization exec default local`	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 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** Beginning from Cisco IOS XE Dublin 17.10.1, 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
```

Additional References for Secure Copy

Related Documents

Related Topic	Document Title
Secure Shell Version 1 and 2 support	*Configuring Secure Shell*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

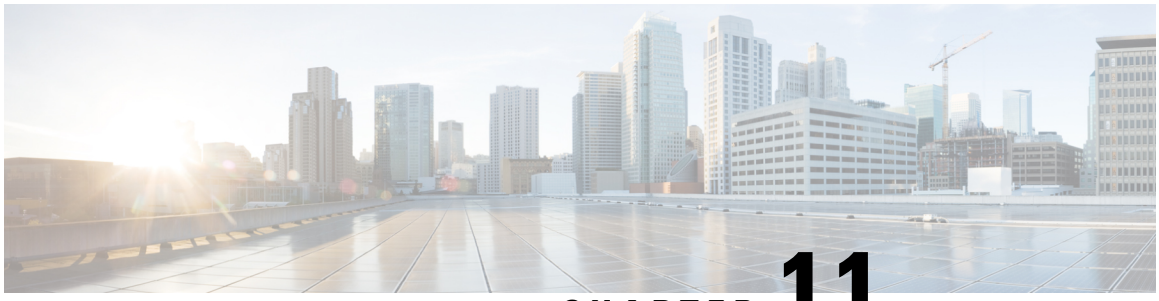
Feature History for Secure Copy

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	Secure Copy	The Secure Copy feature provides a secure and authenticated method for copying device configurations or device image files. SCP relies on SSH, an application and protocol that provide a secure replacement for the Berkeley r-tools suite. The following commands were introduced or modified: **debug ip scp** and **ip scp server enable**.
Cisco IOS XE Amsterdam 17.2.1	Secure Copy Performance Improvements	SSH bulk mode enables certain optimizations to enhance the throughput performance of procedures involving large amount of data transfer. This mode can be enabled by using the **ip ssh bulk-mode** global configuration command.
Cisco IOS XE Bengaluru 17.6.1	Secure Copy Improvement in Large RTT Scenario	Secure copy in large RTT settings can be configured by using the *window-size* variable option of the **ip ssh bulk-mode** command.
Cisco IOS XE Cupertino 17.9.1	Secure Copy	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.
Cisco IOS XE Dublin 17.10.1	Secure Copy Performance Improvements	SSH bulk mode is enabled by default with the default window size of 128KB.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 11

Configuration Replace and Configuration Rollback

- Prerequisites for Configuration Replace and Configuration Rollback, on page 207
- Restrictions for Configuration Replace and Configuration Rollback, on page 208
- Information About Configuration Replace and Configuration Rollback, on page 208
- How to Use Configuration Replace and Configuration Rollback, on page 211
- Configuration Examples for Configuration Replace and Configuration Rollback, on page 217
- Additional References for Configuration Replace and Configuration Rollback, on page 220
- Feature History for Configuration Replace and Configuration Rollback, on page 220

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 ethernet 0** command line from the current running configuration if that interface is physically present on the device. Similarly, the **interface ethernet 1** 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
```

Additional References for Configuration Replace and Configuration Rollback

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9200 Series Switches)*

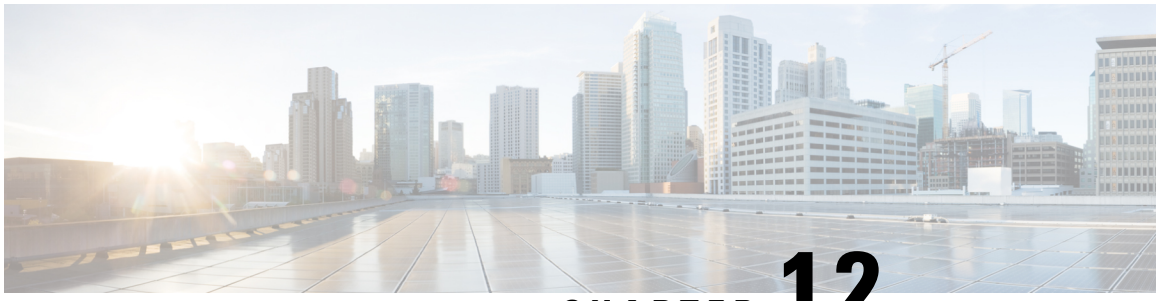
Feature History for Configuration Replace and Configuration Rollback

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	Configuration Replace and Configuration Rollback	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.
Cisco IOS XE Cupertino 17.9.1	Configuration Replace and Configuration Rollback	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 12

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, on page 221
- Information About Software Maintenance Upgrade, on page 221
- How to Manage Software Maintenance Updates, on page 222
- Configuration Examples for Software Maintenance Upgrade, on page 225
- Additional References for Software Maintenance Upgrade, on page 238
- Feature History for Software Maintenance Upgrade, on page 238

Restrictions for Software Maintenance Upgrade

- Hot patching is not supported on Cisco Catalyst 9200 Series Switches.
- SMU supports cold patching using install mode only.
- Prior to Cisco IOS XE Bengaluru 17.9.1, SMU installation was supported both in bundle and install modes. From Cisco IOS XE Bengaluru 17.9.1, 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/cat9k_iosxe-lni.BLD_POLARIS_DEV_LATEST_20210616_160027.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 `cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin` is compatible with software image `cat9k_lite_iosxe.16.09.04.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:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.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 `cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin` is compatible with software image `cat9k_lite_iosxe.16.09.04.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:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin`
	`Device# install add file flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.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:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin, cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.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:cat9k_lite_iosxe.16.09.04.CSCvk70181.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:cat9k_lite_iosxe.16.09.04.CSCvk70181.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.

- Example: Installing an SMU (3-Step Process, Using flash:), on page 226
- Example: Installing Multiple SMUs (3-Step Process, Using flash:), on page 229
- Example: Installing an SMU (3-Step Process, Using TFTP), on page 234
- Example: Managing a SMU Package (Additional show commands, Rollback, Deactivation), on page 236

Example: 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:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
install_add: START Wed Jun 10 14:17:45 IST 2020
install_add: Adding SMU

--- Starting initial file syncing ---
Info: Finished copying flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin to the
selected switch(es)
Finished initial file syncing


*Jun 10 14:17:48.128 IST: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
 Started install add flash:cat9k_lite_iosxe.16.09.04.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/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin Wed Jun 10
 14:18:00 IST 2020
```

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:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
IMG   C    16.9.4.0.3431

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

2. Activating the SMU package file.

```
Device# install activate file flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

install_activate: START Wed Jun 10 14:19:59 IST 2020
install_activate: Activating SMU

*Jun 10 14:20:01.513 IST: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
 Started install activate flash:cat9k_lite_iosxe.16.09.04.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!

*Jun 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
Jun 10 14:20:28.291: %PMAN-5-EXITACTION: F0/0: pvp: Process manager is exiting: reload
fp action requested
Jun 10 14:20:30.718: %PMAN-5-EXITACTION: R0/0: pvp: Proce
Jun 10 14:20:34.834: %PMAN-5-EXITACTION: C0/0: pvp: Process manager is exiting:
Jun 10 14:20:36.053: %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
 install activate SMU flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
watchdog watchdog0: watchdog did not stop!
reboot: Restarting system

Initializing Hardware...
<output truncated>

############
Jun 10 08:52:01.806: %BOOT-5-BOOTTIME_SMU_TEMP_ACTIVE_DETECTED: R0/0: install_engine:
SMU file /flash/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin active temporary...
SMU commit is pending

Cisco IOS Software [Fuji], Catalyst L3 Switch Software (CAT9K_LITE_IOSXE), Version 16.9.4,
 RELEASE SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2019 by Cisco Systems, Inc.
Compiled Thu 22-Aug-19 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:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
IMG   C    16.9.4.0.3431

--------------------------------------------------------------------------------
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 Jun 10 14:38:42 IST 2020
install_commit: Committing SMU
```

```
*Jun 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/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin Wed Jun
 10 14:38:58 IST 2020
*Jun 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:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
IMG   C    16.9.4.0.3431

--------------------------------------------------------------------------------
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:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
IMG   C    16.9.4.0.3431
```

Checking the version, by using the **show version** command:

```
Device# show version
Cisco IOS XE Software, Version 16.09.04
Cisco IOS Software [Fuji], Catalyst L3 Switch Software (CAT9K_LITE_IOSXE), Version 16.9.4,
 RELEASE SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2019 by Cisco Systems, Inc.
Compiled Thu 22-Aug-19 17:30 by mcpre
...
```

Example: Installing Multiple SMUs (3-Step Process, Using flash:)

The following example shows how to install multiple SMU package files by using the 3-step process. Here the SMU package files are saved in the device's flash.

The SMU files being installed on the switch stack are:
`cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin` and
`cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin`

1. (Optional) Checking that the switch stack is ready and that the SMU package files are in the device's flash.

```
Device# show switch
Switch/Stack Mac Address : 08ec.f586.aa80 - Local Mac Address
Mac persistency wait time: Indefinite
                                             H/W   Current
Switch#   Role    Mac Address     Priority Version  State
-----------------------------------------------------------------------------------
*1       Active   08ec.f586.aa80     1      V01     Ready
 2       Member   7488.bb3c.f600     1      V01     Ready
 3       Member   7488.bb3f.9c00     1      V01     Ready
 4       Member   08ec.f5ee.1080     1      V01     Ready
 5       Standby  08ec.f589.7c80     1      V01     Ready

Device# dir flash: | i smu

89075  -rw-  79256 Oct 26 2035 07:07:42 +00:00
cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
89082  -rw-  9656  Oct 26 2035 07:08:08 +00:00
cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
```

2. Copying the SMU package files from flash and adding them.

   Only one SMU package file is added at a time; no reload is required between the addition of the SMU package files.

```
Device# install add file flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
install_add: START Fri Oct 26 07:10:59 UTC 2035
Oct 26 07:11:01.695 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 add flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
install_add: Adding SMU
install_add: Checking whether new add is allowed ....

--- Starting initial file syncing ---

*Oct 26 07:11:01.643: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
Started install add flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin[1]: Copying
flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin from switch 1 to switch 2 3 4 5
[2 3 4 5]: Finished copying to switch 2 switch 3 switch 4 switch 5
Info: Finished copying flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.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
  [3] SMU_ADD package(s) on switch 3
  [3] Finished SMU_ADD on switch 3
  [4] SMU_ADD package(s) on switch 4
```

```
  [4] Finished SMU_ADD on switch 4
  [5] SMU_ADD package(s) on switch 5
  [5] Finished SMU_ADD on switch 5
Checking status of SMU_ADD on [1 2 3 4 5]
SMU_ADD: Passed on [1 2 3 4 5]
Finished SMU Add operation

SUCCESS: install_add  Fri Oct 26 07:11:45 UTC 2035
Oct 26 07:11:46.695 %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
install add SMU flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
Device#
*Oct 26 07:11:46.656: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 1 R0/0: install_engine:
 Completed install add SMU flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
```

Verifying the additiong of the first SMU package file by using the **show install summary** command.

```
Device# show install summary
[ Switch 1 2 3 4 5 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
IMG   C    16.12.3.0.3752

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

Adding the second SMU package file.

```
Device# install add file flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin

install_add: START Fri Oct 26 07:12:38 UTC 2035
Oct 26 07:12:40.782 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 add flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
install_add: Adding SMU
install_add: Checking whether new add is allowed ....

--- Starting initial file syncing ---

*Oct 26 07:12:40.743: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
Started install add flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin[1]: Copying
flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin from switch 1 to switch 2 3 4 5
[2 3 4 5]: Finished copying to switch 2 switch 3 switch 4 switch 5
Info: Finished copying flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.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
  [3] SMU_ADD package(s) on switch 3
  [3] Finished SMU_ADD on switch 3
  [4] SMU_ADD package(s) on switch 4
  [4] Finished SMU_ADD on switch 4
  [5] SMU_ADD package(s) on switch 5
  [5] Finished SMU_ADD on switch 5
Checking status of SMU_ADD on [1 2 3 4 5]
```

```
SMU_ADD: Passed on [1 2 3 4 5]
Finished SMU Add operation

SUCCESS: install_add  Fri Oct 26 07:13:24 UTC 2035
Oct 26 07:13:25.656 %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
install add SMU flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
 Decive#
*Oct 26 07:13:25.616: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 1 R0/0: install_engine:
 Completed install add SMU flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
```

Verifying the addition and installation of both the SMU package files by using the **show install summary** command. The status of both package files is `I`, because they have not been activated and committed yet.

```
Device# show install summary

[ Switch 1 2 3 4 5 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
SMU   I    flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
IMG   C    16.12.3.0.3752

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

3. Activating the SMU package files.

   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.

```
Device# install activate file
flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin,cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin


install_activate: START Sun Oct 28 13:23:42 UTC 2035
Oct 28 13:23:44.620 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 activate
flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin,cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
install_activate: Activating SMU

*Oct 28 13:23:44.581: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
Started install activate
flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin,cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin

This operation may require 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

*Oct 28 13:24:41.563: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 1 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 secondsOct 28 13:24:43.259:
 %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: R0/0: rollback_timer: Install auto abort
 timer will expire in 7200 seconds
*Oct 28 13:24:43.222: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 4 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds
```

```
*Oct 28 13:24:43.192: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 3 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds
*Oct 28 13:24:43.134: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 2 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds
*Oct 28 13:24:43.825: %INSTALL-5-INSTALL_AUTO_ABORT_TIMER_PROGRESS: Switch 5 R0/0:
rollback_timer: Install auto abort timer will expire in 7200 seconds  [1] SMU_ACTIVATE
package(s) on switch 1
  [1] Finished SMU_ACTIVATE on switch 1
  [2] SMU_ACTIVATE package(s) on switch 2
  [2] Finished SMU_ACTIVATE on switch 2
  [3] SMU_ACTIVATE package(s) on switch 3
  [3] Finished SMU_ACTIVATE on switch 3
  [4] SMU_ACTIVATE package(s) on switch 4
  [4] Finished SMU_ACTIVATE on switch 4
  [5] SMU_ACTIVATE package(s) on switch 5
  [5] Finished SMU_ACTIVATE on switch 5
Checking status of SMU_ACTIVATE on [1 2 3 4 5]
SMU_ACTIVATE: Passed on [1 2 3 4 5]
Finished SMU Activate operation

install_activate: Reloading the box to complete activation of the SMU...
install_activate will reload the system now!

Chassis 4 reloading, reason - Reload command
reload fp action requested
rp processes exit with reload switch code



watchdog watchdog0: watchdog did not stop!
reboot: Restarting system

Initializing Hardware...

System Bootstrap, Version 16.12.1r [FC6], RELEASE SOFTWARE (P)
Compiled Thu 02/13/2020 12:36:08 by rel

Current ROMMON image : Primary
C9200L-24T-4G platform with 2097152 Kbytes of main memory

boot: attempting to boot from [flash:packages.conf]
boot: reading file packages.conf

##############
Oct 28 13:26:55.653: %BOOT-5-BOOTTIME_SMU_TEMP_ACTIVE_DETECTED: R0/0: install_engine:
SMU file /flash/cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin active temporary... SMU
 commit is pending
Oct 28 13:26:55.912: %BOOT-5-BOOTTIME_SMU_TEMP_ACTIVE_DETECTED: R0/0: install_engine:
SMU file /flash/cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin active temporary... SMU
 commit is pending

Waiting for 120 seconds for other switches to boot
##########
Switch number is 4
All switches in the stack have been discovered. Accelerating discovery
```

Verifying activation of the SMU package files by using the **show install summary** command. The status of both files is U, because they have not been committed yet.

```
Device# show install summary
[ Switch 1 2 3 4 5 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
```

```
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   U    flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
SMU   U    flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
IMG   C    16.12.3.0.3752

--------------------------------------------------------------------------------
Auto abort timer: active on install_activate, time before rollback - 01:50:16
--------------------------------------------------------------------------------
```

4. Committing the SMU package file

```
Device# install commit
install_commit: START Sun Oct 28 13:34:42 UTC 2035
Oct 28 13:34:45.202 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 commit

*Oct 28 13:34:45.146: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0: install_engine:
Started install commitinstall_commit: Committing SMU
Executing pre scripts....
Executing pre sripts done.
--- Starting SMU Commit operation ---
Performing SMU_COMMIT on all members

*Oct 28 13:35:24.436: %PLATFORM-4-ELEMENT_WARNING: Switch 1 R0/0: smand: 5/RP/0: limited
 space - copy files out of flash: directory. flash: value 84% (1599 MB) exceeds warning
 level 70% (1337 MB).
*Oct 28 13:35:30.587: %PLATFORM-4-ELEMENT_WARNING: Switch 1 R0/0: smand: 2/RP/0: limited
 space - copy files out of flash: directory. flash: value 74% (1412 MB) exceeds warning
 level 70% (1337 MB).  [1] SMU_COMMIT package(s) on switch 1
  [1] Finished SMU_COMMIT on switch 1
  [2] SMU_COMMIT package(s) on switch 2
  [2] Finished SMU_COMMIT on switch 2
  [3] SMU_COMMIT package(s) on switch 3
  [3] Finished SMU_COMMIT on switch 3
  [4] SMU_COMMIT package(s) on switch 4
  [4] Finished SMU_COMMIT on switch 4
  [5] SMU_COMMIT package(s) on switch 5
  [5] Finished SMU_COMMIT on switch 5
Checking status of SMU_COMMIT on [1 2 3 4 5]
SMU_COMMIT: Passed on [1 2 3 4 5]
Finished SMU Commit operation

SUCCESS: install_commit /flash/cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
/flash/cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
Sun Oct 28 13:35:52 UTC 2035
Oct 28 13:35:53.789 %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
install commit SMU

JJ22-Vore_stack-24TE#
*Oct 28 13:35:53.749: %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 files have been installed, activated and committed, and the status is C.

```
Device# show install summary
[ Switch 1 2 3 4 5 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
```

```
SMU   C    flash:cat9k_lite_iosxe.16.12.03.CSCvt22238.SPA.smu.bin
SMU   C    flash:cat9k_lite_iosxe.16.12.03.CSCvt72427.SPA.smu.bin
IMG   C    16.12.3.0.3752

--------------------------------------------------------------------------------
Auto abort timer: inactive
--------------------------------------------------------------------------------
```

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/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

Jun 22 11:32:27.035: %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 add tftp://172.16.0.1//tftpboot/folder1/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Jun 22 11:32:27.035 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 add tftp://172.16.0.1//tftpboot/folder1/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Downloading file
tftp://172.16.0.1//tftpboot/folder1/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Finished downloading file
tftp://172.16.0.1//tftpboot/folder1/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin to
 flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
install_add: Adding SMU
install_add: Checking whether new add is allowed ....

--- Starting initial file syncing ---

025335: *Jun 22 2020 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/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin[1]:
 Copying flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin from switch 1 to switch
 2
[2]: Finished copying to switch 2
Info: Finished copying flash:cat9k_lite_iosxe.16.09.04.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 Jun 22 11:32:56 UTC 2020
Jun 22 11:32:57.598: %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
 install add SMU flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Jun 22 11:32:57.598 %INSTALL-5-INSTALL_COMPLETED_INFO: R0/0: install_engine: Completed
install add SMU flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

ECSG-SEC-C9200-24P#
025336: *Jun 22 2020 11:32:57 UTC: %INSTALL-5-INSTALL_COMPLETED_INFO: Switch 1 R0/0:
install_engine: Completed install add SMU
```

```
flash:cat9k_lite_iosxe.16.09.04.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:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
IMG C 16.12.02.0.6

--------------------------------------------------------------------------------
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:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

install_activate: START Mon Jun 22 11:37:17 UTC 2020

Jun 22 11:37:37.582: %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 activate flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Jun 22 11:37:37.582 %INSTALL-5-INSTALL_START_INFO: R0/0: install_engine: Started install
 activate flash:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
install_activate: Activating SMU

025337: *Jun 22 2020 11:37:37 UTC: %INSTALL-5-INSTALL_START_INFO: Switch 1 R0/0:
install_engine: Started install activate
flash:cat9k_lite_iosxe.16.09.04.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 16.09.04
Cisco IOS Software [Fuji], Catalyst L3 Switch Software (CAT9K_LITE_IOSXE), Version 16.9.4,
 RELEASE SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2019 by Cisco Systems, Inc.
Compiled Thu 22-Aug-19 17:30 by mcpre
<output truncated>
```

3. Committing the SMU package file.

```
Device# install commit

install_commit: START Mon Jun 22 11:38:48 UTC 2020
SUCCESS: install_commit Mon Jun 22 11:38:52 UTC 2020
Device#
```

Verifying that the update package is now committed, and that it will be persistent across reloads:

```
Device# show install summary

Active Packages:
tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Inactive Packages:
No packages
Committed Packages:
tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Uncommitted Packages:
No packages
Device#
```

Example: Managing a SMU Package (Additional show commands, Rollback, Deactivation)

The following sample output displays information about active, inactive, committed, and uncommitted packages by using the **show install summary** command. Here SMU package file `cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin` is active and committed:

```
Device# show install summary

Active Packages:
    tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Inactive Packages:
   No packages
Committed Packages:
    tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Uncommitted Packages:
   No packages
Device#
```

The following is sample output from the **show install active** command:

```
Device# show install active

Active Packages:
tftp:cat3k-universalk9.2017-01-10_13.15.1.CSCxxx.SSA.dmp.bin
```

The following example shows how to rollback an update package to the committed package:

```
Device# install rollback to base

install_rollback: START Wed Jun 10 11:27:41 IST 2020
This rollback would require a reload. Do you want to proceed? [y/n]y
2 install_rollback: Reloading the box to take effect

Initializing Hardware ...
<after reload>
Device#
```

The following is sample output from the **show install summary** command:

```
Device# show install summary

Active Packages:
tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Inactive Packages:
No packages
Committed Packages:
tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
```

```
Uncommitted Packages:
No packages
Device#
```

The following is sample output from the **show install log** command:

```
Device# show install log

[0|install_op_boot]: START Wed Jun 10 19:31:50 Universal 2020
[0|install_op_boot]: END SUCCESS Wed Jun 10 19:31:56 Universal 2020
```

The following example shows how to deactivate an SMU package file:

```
Device# install deactivate file tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

install_deactivate: START Wed Jun 10 10:49:07 IST 2020
The activation step would require a reload. Do you want to proceed? [y/n]y
Regular SMU. Reloading the box to complete activation of the SMU...

Initializing Hardware...
...
<after reload>
Device#
```

The following is sample output from the **show install summary** command:

```
Device# show install summary

Active Packages:
No packages
Inactive Packages:tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin
Committed Packages:
No packages
Uncommitted Packages:
No packages
Device#
```

The following example shows how to remove an SMU from the device:

```
Device# install remove file tftp:cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin

install_remove: START Wed Jun 10 12:09:43 IST 2020
SUCCESS: install_remove /tftp/cat9k_lite_iosxe.16.09.04.CSCvk70181.SPA.smu.bin Wed Jun 10
12:09:49 IST 2020
Device#
```

The following is sample output from the **show install summary** command:

```
Device# show install summary

Active Packages:
No packages
Inactive Packages:
No packages
Committed Packages:
No packages
Uncommitted Packages:
No packages
```

Additional References for Software Maintenance Upgrade

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9200 Series Switches)*

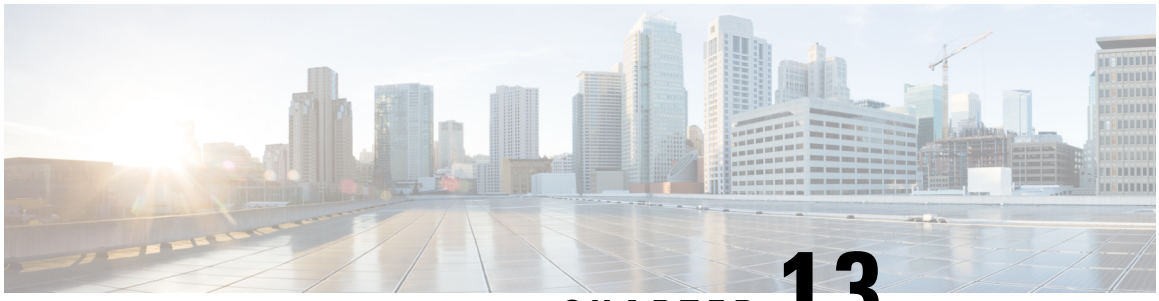
Feature History for Software Maintenance Upgrade

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.4	Software Maintenance Upgrade (SMU)	An SMU is a package that can be installed on a system to provide a fix or a security resolution to a released image. On this platform, SMUs require a cold (complete) reload of the operating system; hot patching is not supported.
Cisco IOS XE Gibraltar 16.10.1	Public Key Infrastructure (PKI) Patching	The SMU package supports patching of the PKI component.
Cisco IOS XE Gibraltar 16.12.1	Software Maintenance Upgrade (SMU)	Support for this feature was introduced on the C9200 SKUs. Hot patching is not supported.
Cisco IOS XE Cupertino 17.9.1	Software Maintenance Upgrade (SMU)	SMU installation is supported in install mode only.
	Software Maintenance Upgrade (SMU)	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 13

Working with the Flash File System

- Information About the Flash File System, on page 239
- Displaying Available File Systems, on page 239
- Setting the Default File System, on page 242
- Displaying Information About Files on a File System, on page 242
- Changing Directories and Displaying the Working Directory , on page 243
- Creating Directories , on page 244
- Copying Files, on page 244
- Creating, Displaying and Extracting Files , on page 247
- Additional References for Flash File System, on page 249
- Feature History for Flash File System, on page 249

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
```

This example shows a device stack. In this example, the active device is stack member 2; the file system on stack member 1 is displayed as flash-1:,the file system on stack member 2 is displayed as flash-2:, the file system on stack member 3 is displayed as flash-3: and so on up to . The example also shows the crashinfo directories and a USB flash drive plugged into the active device:

```
Device# show file systems
File Systems:

       Size(b)       Free(b)      Type  Flags  Prefixes
             -             -    opaque     rw   system:
             -             -    opaque     rw   tmpsys:
    1651314688    1565089792      disk     rw   crashinfo: crashinfo-2:
    1651507200    1560281088      disk     rw   crashinfo-1:
    1651507200    1562378240      disk     rw   crashinfo-3: stby-crashinfo:
*  11353194496   10735611904      disk     rw   flash: flash-2:
   11353980928   10152312832      disk     rw   flash-1:
   11353980928    2161115136      disk     rw   flash-3: stby-flash:
   15243046912   14423638016      disk     rw   usbflash0: usbflash0-2:
     520093696     520093696      disk     rw   usbflash0-1:
    3497074688    3417554944      disk     ro   webui:
             -             -    opaque     rw   null:
             -             -    opaque     ro   tar:
             -             -   network     rw   tftp:
       2097152       2085334     nvram     rw   nvram:
             -             -   network     rw   rcp:
             -             -   network     rw   http:
             -             -   network     rw   ftp:
             -             -   network     rw   scp:
             -             -   network     rw   https:
             -             -    opaque     ro   cns:
   21003628544   19867037696      disk     rw   usbflash1: usbflash1-2:
  118014083072  111933390848      disk     rw   usbflash1-3: stby-usbflash1:
       2097152       2085334     nvram     rw   stby-nvram:
             -             -     nvram     rw   stby-rcsf:
             -             -    opaque     rw   revrcsf:
```

Table 9: 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.
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 10: 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:** `Device# mkdir new_configs`	Creates a new directory. Directory names are case sensitive and are limited to 45 characters between the slashes (/); the name cannot contain 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

Copying Files from One Device in a Stack to Another Device in the Same Stack

To copy a file from one device in a stack to another device in the same stack, use the **flash-X:** notation, where **X** is the device number.

To view all devices in a stack, use the **show switch** command in privileged EXEC mode, as in the following example of a 9-member device stack:

```
Device# show switch
Switch/Stack Mac Address : 0006.f6b9.b580 - Local Mac Address Mac persistency wait time:
Indefinite
                                             H/W   Current
Switch#   Role    Mac Address     Priority Version  State
-----------------------------------------------------------
*1       Active   0006.f6b9.b580     15     P3B     Ready
 2       Standby  0006.f6ba.0c80     14     P3B     Ready
 3       Member   0006.f6ba.3300     7      P3B     Ready
 4       Member   0006.f6b9.df80     6      P3B     Ready
 5       Member   0006.f6ba.3880     13     P1A     Ready
 6       Member   1ce6.c7b6.ef00     4      PP      Ready
 7       Member   2037.06ce.2580     3      P2A     Ready
 8       Member   2037.0653.7e00     2      P5A     Ready
 9       Member   2037.0653.9280     1      P5B     Ready
```

To view all devices in a stack, use the **show switch** command in privileged EXEC mode, as in the following example of a 8-member device stack:

```
Device# show switch
Switch/Stack Mac Address : 046c.9d01.3b80 - Local Mac Address
Mac persistency wait time: 4 mins
                                                     H/W   Current
Switch#   Role    Mac Address     Priority Version  State
------------------------------------------------------------
*1       Active   046c.9d01.3b80     15     P4B     Ready
 2       Standby  046c.9d01.0f80     13     P3C     Ready
 3       Member   046c.9d01.1180     11     P4B     Ready
 4       Member   046c.9d01.0e80     9      P3C     Ready
 5       Member   046c.9d01.4d00     7      P3C     Ready
 6       Member   046c.9d01.2800     5      P3C     Ready
 7       Member   046c.9d01.6e80     3      P4B     Ready
 8       Member   046c.9d01.8180     1      P4B     Ready
```

To view all file systems available to copy on a specific device, use the **copy** command as in the following example of a 5-member stack:

```
Device# copy flash:?
flash:.installer
flash:.prst_sync
flash:.rollback_timer
flash:boothelper.log
flash:bootloader_evt_handle.log
flash:cat9k-cc_srdriver.16.05.01a.SPA.pkg
flash:cat9k-espbase.16.05.01a.SPA.pkg
flash:cat9k-guestshell.16.05.01a.SPA.pkg
flash:cat9k-rpbase.16.05.01a.SPA.pkg
flash:cat9k-rpboot.16.05.01a.SPA.pkg
flash:cat9k-sipbase.16.05.01a.SPA.pkg
flash:cat9k-sipspa.16.05.01a.SPA.pkg
flash:cat9k-srdriver.16.05.01a.SPA.pkg
flash:cat9k-webui.16.05.01a.SPA.pkg
flash:cat9k-wlc.16.05.01a.SPA.pkg
flash:core
flash:dc_profile_dir
flash:dc_stats.txt
flash:gs_script
flash:nvram_config
flash:packages.conf
```

This example shows how to copy a config file stored in the flash partition of device 2 to the flash partition of device 4. It assumes that device 2 and device 4 are in the same stack.

```
Device# copy flash-2:config.txt flash-4:config.txt
```

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* **Example:** `Device# archive tar /create tftp:172.20.10.30/saved. flash:/new-configs`	Creates a file and adds files to it. 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.

	Command or Action	Purpose
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 tftp:/172.20.10.30/saved. flash:/new-configs`	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 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:**	Displays the contents of any readable file, including a file on a remote file system.

	Command or Action	Purpose
	`Device# more flash:/new-configs`	

Additional References for Flash File System

Related Documents

Related Topic	Document Title
Commands for managing flash: file systems	*Cisco IOS Configuration Fundamentals Command Reference*

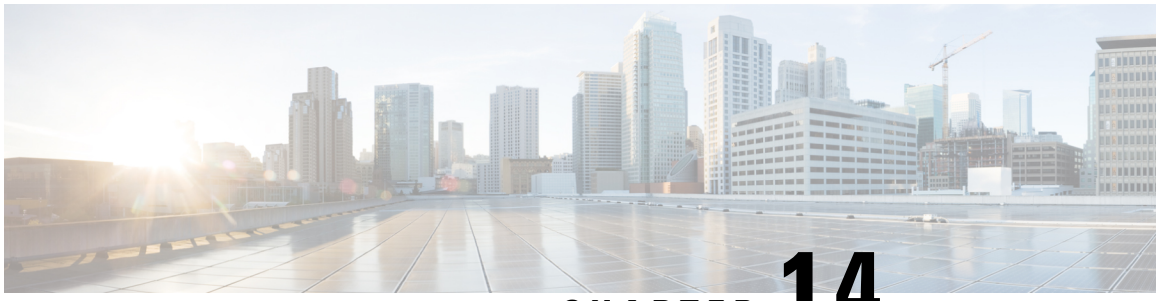
Feature History for Flash File System

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	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.
Cisco IOS XE Cupertino 17.9.1	Flash File System	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 14

Performing Factory Reset

- Prerequisites for Performing a Factory Reset, on page 251
- Restrictions for Performing a Factory Reset, on page 251
- Information About Performing a Factory Reset, on page 251
- How to Perform a Factory Reset, on page 253
- Configuration Examples for Performing a Factory Reset, on page 254
- Additional References for Performing a Factory Reset, on page 256
- Feature History for Performing a Factory Reset, on page 256

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 11: 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. NIST 800-88 is a standard published by the National Institute of Standards and Technology (NIST) that provides guidelines for media sanitization. The PURGE standard within NIST 800-88 specifies methods to render data on storage media unrecoverable using laboratory techniques. When a device is sanitized using

NIST 800-88 PURGE method, data cannot be recovered through simple non-invasive data recovery techniques or advanced laboratory techniques.

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**}
	• For stacked devices:
	factory-reset {**all** [**secure 3-pass**] | **config** | **boot-vars** | **switch** {*switch-number* | **all** {**all** [**secure 3-pass**] | **config** | **boot-vars**}}
	Example:
	`Device# factory-reset all`
	OR
	`Device# factory-reset switch 1 all config`
	OR
	`Device# factory-reset all secure`
		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.

	Command or Action	Purpose

		• **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.
		  • 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.
		• **switch** {*switch-number* | **all**}:
		  • *switch-number*: Specifies the switch number. The range is from 1 to 16.
		  • **all**: Selects all the switches in the stack.
		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 examples show how to perform a factory reset on stacked devices:

```
Device> enable
Device# factory-reset switch all 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]
Chassis 1 reloading, reason - Factory Reset

 Protection key not found
9300L#Oct 25 09:53:05.740: %PMAN-5-EXITACTION: F0/0: pvp: Process manager is exiting: reload
 fp action requested
Oct 25 09:53:07.277: %PMAN-5-EXITACTION:vp: Process manager is exiting: rp processes exit
with reload switch code


Enabling factory reset for this reload cycle
 Switch booted with
tftp://10.5.40.45/cat9k_iosxe.BLD_POLARIS_DEV_LATEST_20191007_224933_V17_2_0_21_2.SSA.bin
 Switch booted via
//10.5.40.45/cat9k_iosxe.BLD_POLARIS_DEV_LATEST_20191007_224933_V17_2_0_21_2.SSA.bin
% FACTORYRESET - Started Cleaning Up...

% FACTORYRESET - Unmounting sd1
% FACTORYRESET - Cleaning Up sd1 [0]
% FACTORYRESET - erase In progress.. please wait for completion...
% FACTORYRESET - write zero...
% FACTORYRESET - finish erase

% FACTORYRESET - Making File System sd1 [0]
Discarding device blocks: done
Creating filesystem with 409600 4k blocks and 102544 inodes
Filesystem UUID: fcf01664-7c6f-41ce-99f0-6df1d941701e
Superblock backups stored on blocks:
 32768, 98304, 163840, 229376, 294912

Allocating group tables: done
Writing inode tables: done
Writing superblocks and filesystem accounting information: done

% FACTORYRESET - Mounting Back sd1 [0]
% FACTORYRESET - Handling Mounted sd1
% FACTORYRESET - Factory Reset Done for sd1

% FACTORYRESET - Unmounting sd3
% FACTORYRESET - Cleaning Up sd3 [0]
% FACTORYRESET - erase In progress.. please wait for completion...
% FACTORYRESET - write zero...
```

```
Chassis 2 reloading, reason - Factory Reset
Dec 12 01:02:12.500: %PMAN-5-EXITACTION: F0/0: pvp: Process manager is exiting: reload fp
action requested
De
Enabling factory reset for this reload cycle
 Switch booted with
tftp://10.5.40.45/cat9k_iosxe.BLD_POLARIS_DEV_LATEST_20191007_224933_V17_2_0_21_2.SSA.bin
 Switch booted via
//10.5.40.45/cat9k_iosxe.BLD_POLARIS_DEV_LATEST_20191007_224933_V17_2_0_21_2.SSA.bin
% FACTORYRESET - Started Cleaning Up...
% FACTORYRESET - Unmounting sd1
% FACTORYRESET - Cleaning Up sd1 [0]
% FACTORYRESET - erase In progress.. please wait for completion...
% FACTORYRESET - write zero...


After this the switch will come to boot prompt. Then the customer has to boot the device
from TFTP.
```

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 C9200 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
```

Additional References for Performing a Factory Reset

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	Command Reference

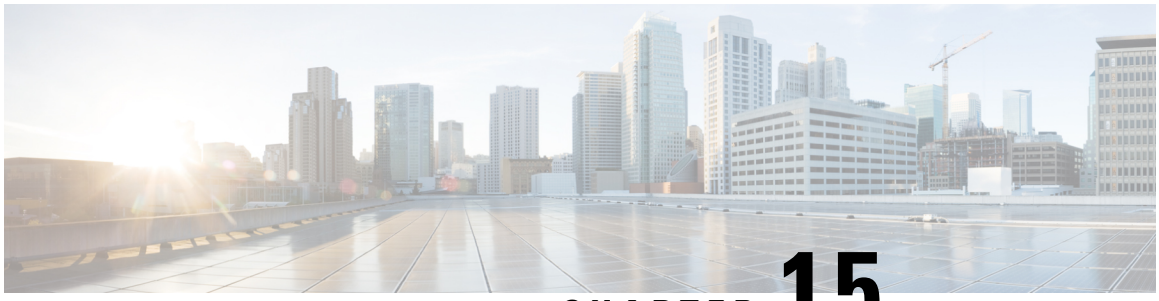
Feature History for Performing a Factory Reset

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	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
Cisco IOS XE Gibraltar 16.12.1	Factory Reset for Removable Storage Devices	Performing a factory reset erases the contents of removable storage devices, such as SATA, SSD, or USB.
Cisco IOS XE Amsterdam 17.2.1	Factory Reset with 3-pass Overwrite	A factory reset can be performed to erase all the content from the device securely with 3-pass overwrite. The **secure 3-pass** keyword was introduced.
	Enhanced Factory Reset Option for Stack and Cisco StackWise Virtual	Support for factory reset on stacked devices and for Cisco StackWise Virtual enabled devices is introduced.
Cisco IOS XE Cupertino 17.9.1	Factory Reset	This feature was implemented on C9200CX-12P-2X2G, C9200CX-8P-2X2G, and C9200CX-12T-2X2G models of the Cisco Catalyst 9200CX Series Switches, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 15

Configuring Secure Storage

- Information About Secure Storage, on page 259
- Enabling Secure Storage , on page 259
- Disabling Secure Storage , on page 260
- Verifying the Status of Encryption, on page 260
- Feature History for Secure Storage, on page 261

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:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
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.

```
Device#show parser encrypt file status
Feature: Enabled
File Format: Plain Text
Encryption Version: Ver1
```

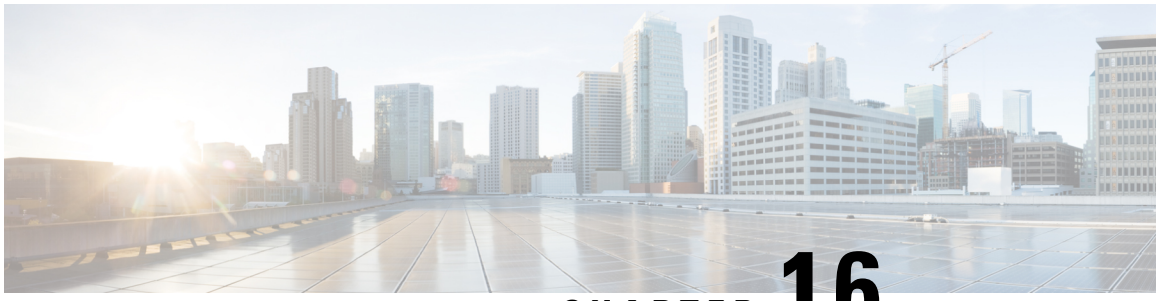
Feature History for Secure Storage

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	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.

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 16

Trace Management

- Information About Trace Management, on page 263
- How to Configure Conditional Debugging, on page 266
- Configuration Examples for Trace Management, on page 269
- Additional References for Trace Management, on page 272
- Feature History for Trace Management, on page 272

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 12: 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.

Tracing Level	Description
Debug	The message provides debug-level output.
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 13: 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 14: 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.

	Command or Action	Purpose
Step 5	**debug platform condition stop** **Example:** `Device# debug platform condition stop`	Stops conditional debugging (this step stops radioactive tracing).
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: [17.11.01], 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: [17.12.01], 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,wncloudn,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: [17.12.01], 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
============================================================
```

Additional References for Trace Management

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	Command Reference Guide for catalyst 9K platforms.

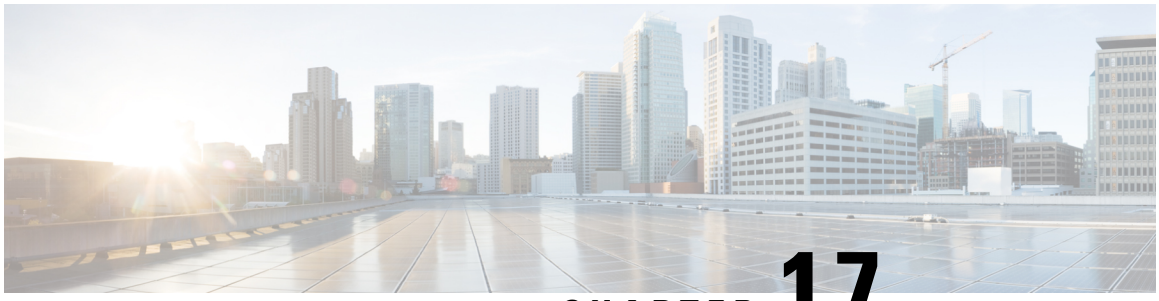
Feature History for Trace Management

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	Conditional Debugging and Radioactive Tracing	The Conditional Debugging feature allows you to selectively enable debugging and logging for specific features based on the set of conditions you define.
Cisco IOS XE Cupertino 17.7.x	Binary Tracing	Binary tracing helps in gathering of trace information with a minimal impact on performance.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 17

Consent Token

- Restrictions for Consent Token, on page 273
- Information About Consent Token, on page 273
- Consent Token Authorization Process for System Shell Access, on page 274
- Feature History for Consent Token, on page 275

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.

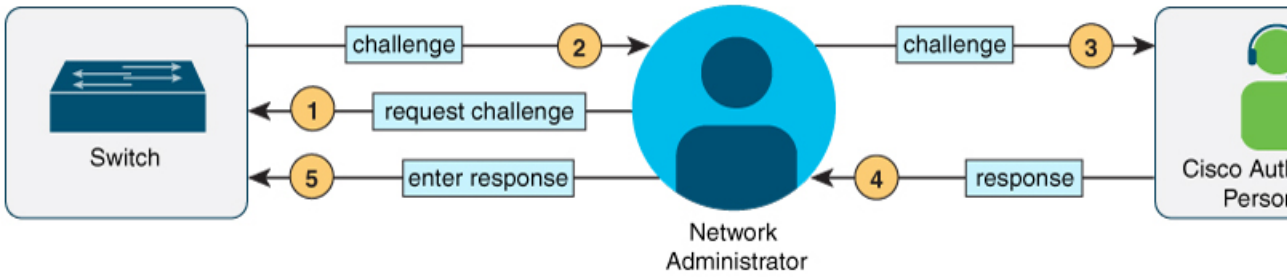
When you request access to system shell, you need to be authorized. You must first run the command to generate a challenge using the Consent Token feature on your device. The device generates a unique challenge as output. You must then copy this challenge string and send it to a Cisco Authorized Personnel 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 3: 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
zSdrAAAQEBAQAAAEAgEAAAAAMAH86csUmDl0BAQ0Fvd7CxqRUeD7B4wW7QUBAAAG8GAhDVEFFEVNTwcAGNUQ9ERU1PXNUQ9TSUdOSU5HX0tEWQgAC0M5OAwLNMLu5CQALOjpQLEVESE5RkI=
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
[redacted]
% 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.

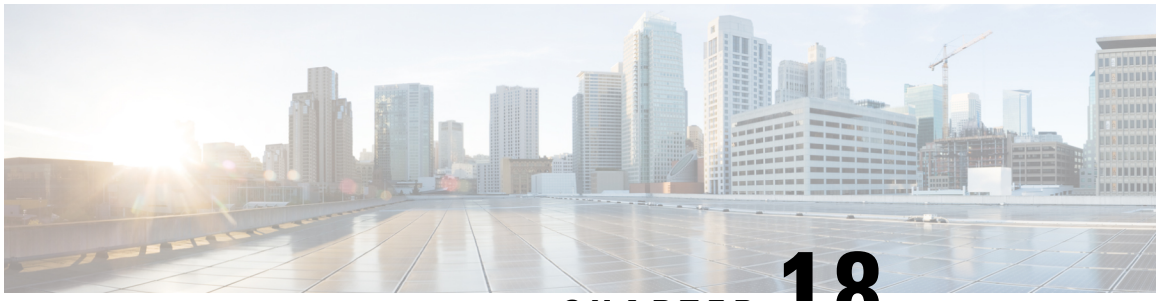
Feature History for Consent Token

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Gibraltar 16.11.1	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).

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 18

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, on page 277
- How to Troubleshoot the Software Configuration, on page 283
- Verifying Troubleshooting of the Software Configuration, on page 291
- Scenarios for Troubleshooting the Software Configuration, on page 292
- Configuration Examples for Troubleshooting Software, on page 294
- Additional References for Troubleshooting Software Configuration, on page 296
- Feature History for Troubleshooting Software Configuration, on page 296

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 283 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 289 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 disabled by default and can be enabled by running the **l2 traceroute** command in global configuration mode. To disable Layer 2 traceroute, use the **no 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 295 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)
Switch#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.

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 .
- Power over Ethernet (PoE)—Record of the power consumption of PoE ports on a standalone device .
- Temperature—Temperature of a standalone deicev .
- 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

Cisco Catalyst 9200CX Series Switches 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   cat9k_lite_iosxe.17.09.01.SPA.bin

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

switch:
```

To boot from the recovery partition, enter **boot drec0:<image name>**:

```
switch: boot drec0:cat9k_lite_iosxe.17.09.01.SPA.bin

boot: attempting to boot from [drec0:cat9k_lite_iosxe.17.09.01prd9.SPA.bin]
boot: reading file cat9k_lite_iosxe.17.09.01.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 or the entire switch stack.

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 09/29/2020 18:05:06 by rel

Current ROMMON image : Primary
C9200-24P 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
```

Note
If an error message is displayed, configure the ignore startup command as **set SWITCH_IGNORE_STARTUP_CFG=1** before entering the **SWITCH_IGNORE_STARTUP_CFG=1** command.

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 Temperature

The Device monitors the temperature conditions and uses the temperature information to control the fans.

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 15: 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 16: 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 17: 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
Cisco pre-standard powered device disconnects or resets. After working normally, a Cisco phone intermittently reloads or disconnects from PoE.	Verify all electrical connections from the switch to the powered device. Any unreliable connection results in power interruptions and irregular powered device functioning such as erratic powered device disconnects and reloads. Verify that the cable length is not more than 100 meters from the switch port to the powered device. Notice what changes in the electrical environment at the switch location or what happens at the powered device when the disconnect occurs. Notice whether any error messages appear at the same time a disconnect occurs. Use the **show log** privileged EXEC command to review error messages. Verify that an IP phone is not losing access to the Call Manager immediately before the reload occurs. (It might be a network problem and not a PoE problem.) Replace the powered device with a non-PoE device, and verify that the device works correctly. If a non-PoE device has link problems or a high error rate, the problem might be an unreliable cable connection between the switch port and the powered device.
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 18: 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 19: 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.

Character	Description
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.

Additional References for Troubleshooting Software Configuration

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9200 Series Switches)*

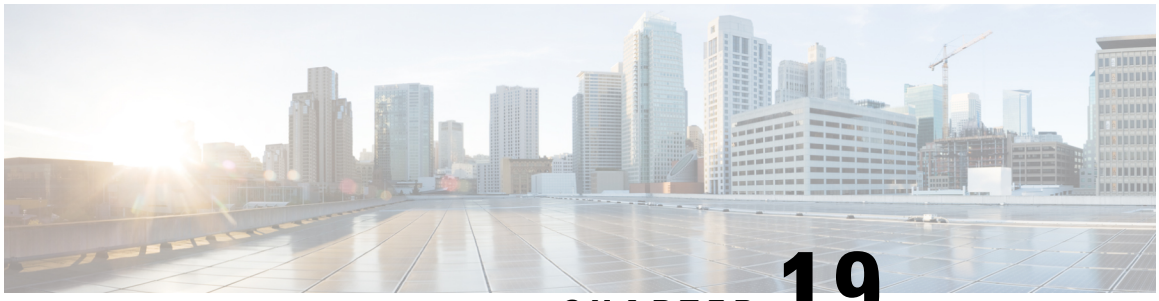
Feature History for Troubleshooting Software Configuration

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Fuji 16.9.2	Troubleshooting Software Configuration	Troubleshooting software configuration describes how to identify and resolve software problems related to the Cisco IOS software on the switch.
Cisco IOS XE Amsterdam 17.3.1	System-Report Files	The hostname is prepended to the system-report files. This makes the system-report files uniquely identifiable.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 19

Line Auto Consolidation

- Line Auto Consolidation, on page 297
- Feature History for Line Auto Consolidation, on page 303

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 Digital Network Architecture (DNA) Center or Cisco vManage and 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 DNA Center. The mismatch in configurations can lead to reverse sync from the device to the DNA Center. The device will be locked from any other configuration changes during this reverse sync. This can affect the performance of the device.

Starting with Cisco IOS XE 17.4.1 release, 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:

Note Certain device configurations and protocols are identified as insecure and pose a security risk to Cisco devices. While existing deployments continue to function, you must intentionally enable these features for new installations. For more information on remediation, refer to Cisco C9000 Switching IOS XE – Resilient Infrastructure Playbook.

- 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
```

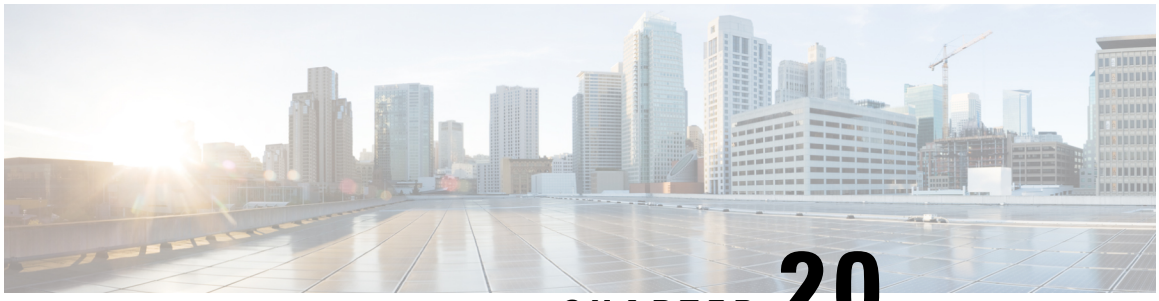
Feature History for Line Auto Consolidation

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Bengaluru 17.4.1	Line Auto Consolidation	Auto Consolidation of Line commands is enabled by default. The **no line auto-consolidation** command can be used to disable the auto consolidation of Line commands. The **line auto-consolidation** command was introduced.

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 20

Troubleshooting System Management

- Overview, on page 305
- Support Articles, on page 305
- Feedback Request (Reference), on page 306
- Disclaimer and Caution (Reference), on page 307

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.

Support Articles

The documents in this section were created using specific software and hardware listed in the Components Used section of each article. However, this does not mean that they are limited to what is listed in Components Used, and generally remain relevant for later versions of software and hardware. Note that there could be some changes in the software or hardware that can cause commands to stop working, the syntax to change, or GUIs and CLIs to look different from one release to another.

The following are the support articles associated with this technology:

Document	Description
Cisco Smart Licensing - Troubleshooting Steps and Considerations on Catalyst platforms	This document describes how to work with Cisco Smart Licensing (cloud-based system) to manage software licenses on Catalyst switches.
Recommended Releases for Catalyst 9200/9300/9400/9500/9600 and Catalyst 3650/3850 Platforms	This document is to help customers find a stable software release for the enterprise switching platforms running Catalyst 9000 series switches.

Document	Description
Migrate Catalyst License to Smart Licensing Using Policy	This document describes what to expect after migration from an older license mechanism to the new "Smart Licensing Using Policy" mechanism in Cisco IOS XE 17.3.2 release and future releases.
Smart Licensing using Policy on Catalyst Switching Platforms	This document describes the Smart Licensing feature using Policy on Catalyst Switching Platforms and its various supported deployment mechanisms, from Cisco IOS XE 17.3.2 release and future releases.
Troubleshoot and Recover Catalyst 9000 Switches from Upgrade Failure Scenarios	This document describes the common failure scenarios that occur when Catalyst 9000 series devices are upgraded along with the procedure to recover them.
Configuration Register equivalent CLIs in IOS-XE	This document describes how to modify certain system parameters using CLI commands on Catalyst 9000 switches running Cisco IOS XE. These commands are an alternative to changing the configuration-register value on Cisco IOS.
Understand Hardware Resources on Catalyst 9000 Switches	This document describes how to understand and troubleshoot hardware resources on Catalyst 9000 series switches.
Understand IPv4 Hardware Resources on Catalyst 9000 Switches	This document describes how to understand and verify IPv4 Forwarding Information Base (FIB) hardware usage on Catalyst 9000 series switches.
Use the -O Option to Ensure Successful SCP from Clients on OpenSSH9.0 to IOS XE Devices	This document describes how to use the -O option to ensure successful SCP from clients on OpenSSH9.0 to Cisco IOS XE devices.

Feedback Request (Reference)

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 (Reference)

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.