IP Addressing Services Configuration Guide, Cisco IOS XE 17.14.x (Catalyst 9600 Switches)

First Published: 2024-04-05

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)

© 2024 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 9600 Series Switches documentation, located at:

  https://www.cisco.com/c/en/us/products/switches/catalyst-9600-series-switches/index.html

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

IP Addressing Services Overview 1

Understanding IPv6 1

IPv6 Addresses 1

128-Bit Wide Unicast Addresses 2

DNS for IPv6 2

IPv6 Stateless Autoconfiguration and Duplicate Address Detection 3

IPv6 Applications 3

DHCP for IPv6 Address Assignment 3

HTTP(S) Over IPv6 3

CHAPTER 2

IPv6 Client IP Address Learning 5

Prerequisites for IPv6 Client Address Learning 5

Information About IPv6 Client Address Learning 5

SLAAC Address Assignment 6

Stateful DHCPv6 Address Assignment 7

Static IP Address Assignment 8

Router Solicitation 8

Router Advertisement 8

Neighbor Discovery 8

Neighbor Discovery Suppression 8

RA Guard 9

How to Configure IPv6 Client Address Learning 9
Configuring IPv6 Unicast 9
Configuring RA Guard Policy 10
Applying RA Guard Policy 11
Configuring IPv6 Snooping 12
Configuring IPv6 ND Suppress Policy 13
Configuring IPv6 Snooping on VLAN/PortChannel 14
Configuring IPv6 on Switch Interface 15
Configuring DHCP Pool on Switch Interface 15
Configuring Stateless Auto Address Configuration Without DHCP 16
Configuring Stateless Auto Address Configuration With DHCP 18
Configuring Stateful DHCP Locally 19
Configuring Stateful DHCP Externally 21
Verifying IPv6 Address Learning Configuration 22
Additional References 23
Feature History for IPv6 Client Address Learning 23

CHAPTER 3 **Configuring DHCP** 25
Prerequisites for Configuring DHCP 25
Restrictions for Configuring DHCP 26
Information About DHCP 26
DHCP Server 26
DHCP Relay Agent 26
DHCP Snooping 27
Option-82 Data Insertion 28
Cisco IOS DHCP Server Database 31
DHCP Snooping Binding Database 31
Default DHCP Snooping Configuration 32
DHCP Snooping Configuration Guidelines 33
DHCP Server Port-Based Address Allocation 33
Default Port-Based Address Allocation Configuration 34
Port-Based Address Allocation Configuration Guidelines 34
How to Configure DHCP 34
Configuring the DHCP Server 34

Configuring the DHCP Relay Agent 34
Specifying the Packet Forwarding Address 35
Configuring DHCP for IPv6 Address Assignment 37
Default DHCPv6 Address Assignment Configuration 37
DHCPv6 Address Assignment Configuration Guidelines 37
Enabling DHCPv6 Server Function (CLI) 37
Enabling DHCPv6 Client Function 40
Enabling the Cisco IOS DHCP Server Database 41
Enabling the DHCP Snooping Binding Database Agent 41
Monitoring DHCP Snooping Information 43
Enabling DHCP Server Port-Based Address Allocation 43
Monitoring DHCP Server Port-Based Address Allocation 44
Feature History for DHCP 45

CHAPTER 4 **DHCP Gleaning 47**
Prerequisites for DHCP Gleaning 47
Information About DHCP Gleaning 47
Overview of DHCP Gleaning 47
DHCP Snooping 48
Configuring an Interface as a Trusted or an Untrusted Source for DHCP Gleaning 48
Example: Configuring an Interface as a Trusted or an Untrusted Source for DHCP Gleaning 49
Additional References for DHCP Gleaning 50
Feature History for DHCP Gleaning 50

CHAPTER 5 **DHCP Options Support 51**
Restrictions for DHCP Options Support 51
Information About DHCP Options Support 51
DHCP Option 82 Configurable Circuit ID and Remote ID Overview 51
DHCP Client Option 12 52
Configuring DHCP Snooping on Private VLANs 52
Example: Mapping Private-VLAN Associations 54
Configuration Examples for DHCP Options Support 55
Feature History for DHCP Options Support 55

CHAPTER 6 **DHCPv6 Options Support** 57

Information About DHCPv6 Options Support 57

CAPWAP Access Controller DHCPv6 Option 57

DNS Search List Option 57

DHCPv6 Client Link-Layer Address Option 58

DHCP Relay Agent 58

DHCPv6 Relay Agent 58

DHCPv6 Relay Interface-Id Option 59

How to Configure DHCPv6 Options Support 59

Configuring CAPWAP Access Points 59

Configuring DNS Search List Using IPv6 Router Advertisement Options 60

Example: Configuring CAPWAP Access Points 61

Verifying DHCPv6 Options Support 62

Additional References for DHCPv6 Options Support 62

Feature History for DHCPv6 Options Support 63

CHAPTER 7 **DHCPv6 Relay Source Configuration** 65

Restrictions for Configuring a DHCPv6 Relay Source 65

Information About DHCPv6 Relay Source Configuration 65

Configuring a DHCPv6 Relay Source 66

Configuring a DHCPv6 Relay Source on an Interface 66

Configuring a DHCPv6 Relay Source Globally 67

Example: Configuring a DHCPv6 Relay Source on an Interface 67

Additional References for DHCPv6 Relay Source Configuration 68

Feature History for DHCPv6 Relay Source Configuration 68

CHAPTER 8 **Configuring GRE IPv6 Tunnels** 69

Restrictions for GRE IPv6 Tunnels 69

Information About GRE IPv6 Tunnels 69

Overview of GRE IPv6 Tunnels 69

GRE IPv6 Tunnel Protection 70

Distributed GRE Tunneling Support 70

How to Configure GRE IPv6 Tunnels 70

Configuring GRE IPv6 Tunnels 70
Configuring GRE IPv6 Tunnel Protection 72
Configuration Examples for GRE IPv6 Tunnels 73
Example: Configuring GRE IPv6 Tunnels 73
Example: Configuring GRE IPv6 Tunnel Protection 73
Feature History for GRE IPv6 Tunnels 74

CHAPTER 9 **Configuring IPv6 over IPv4 GRE Tunnels 75**
Restrictions for IPv6 over IPv4 GRE Tunnels 75
Information About Configuring IPv6 over IPv4 GRE Tunnels 75
Overlay Tunnels for IPv6 75
GRE IPv4 Tunnel Support for IPv6 Traffic 76
Configuring GRE IPv6 Tunnels 76
Configuration Example: Tunnel Destination Address for IPv6 Tunnel 78
Additional References 78
Feature History for IPv6 over IPv4 GRE Tunnels 78

CHAPTER 10 **Configuring GLBP 81**
Restrictions for GLBP 81
Prerequisites for GLBP 81
Information About GLBP 81
GLBP Overview 81
GLBP Active Virtual Gateway 82
GLBP Virtual MAC Address Assignment 83
GLBP Virtual Gateway Redundancy 83
GLBP Virtual Forwarder Redundancy 83
GLBP Gateway Priority 84
GLBP Gateway Weighting and Tracking 84
GLBP MD5 Authentication 85
ISSU-GLBP 85
GLBP SSO 85
GLBP Benefits 86
How to Configure GLBP 86
Customizing GLBP 86

Configuring GLBP MD5 Authentication Using a Key String 89
Configuring GLBP MD5 Authentication Using a Key Chain 90
Configuring GLBP Text Authentication 92
Configuring GLBP Weighting Values and Object Tracking 94
Troubleshooting GLBP 95
Configuration Examples for GLBP 97
Example: Customizing GLBP Configuration 97
Example: Configuring GLBP MD5 Authentication Using Key Strings 97
Example: Configuring GLBP MD5 Authentication Using Key Chains 97
Example: Configuring GLBP Text Authentication 98
Example: Configuring GLBP Weighting 98
Example: Enabling GLBP Configuration 98
Additional References for GLBP 98
Feature History for GLBP 98

CHAPTER 11 **Configuring HSRP** 101
Restriction About Hot Standby Router Protocol 101
Information About Hot Standby Router Protocol 101
HSRP Overview 101
HSRP Versions 103
Multiple HSRP 104
SSO HSRP 104
HSRP and Switch Stacks 105
Configuring HSRP for IPv6 105
HSRP IPv6 Virtual MAC Address Range 105
HSRP IPv6 UDP Port Number 105
How to Configure Hot Standby Router Protocol 105
Default HSRP Configuration 105
HSRP Configuration Guidelines 106
Enabling HSRP 106
Enabling and Verifying an HSRP Group for IPv6 Operation 108
Configuring HSRP Priority 110
Configuring MHSRP 112
Configuring Router A 113

Configuring Router B 116
Configuring HSRP Authentication and Timers 119
Enabling HSRP Support for ICMP Redirect Messages 121
Configuring HSRP Groups and Clustering 121
Verifying HSRP Configurations 121
Configuration Examples for Hot Standby Router Protocol 122
Enabling HSRP: Example 122
Example: Configuration and Verification for an HSRP Group 122
Configuring HSRP Priority: Example 124
Configuring MHSRP: Example 124
Configuring HSRP Authentication and Timer: Example 124
Configuring HSRP Groups and Clustering: Example 125
Additional References for Configuring HSRP 125
Feature History for HSRP 125

CHAPTER 12 **Configuring NHRP** 127
Restrictions About Next Hop Resolution Protocol 127
Information About Next Hop Resolution Protocol 127
NHRP and NBMA Network Interaction 127
Dynamically Built Hub-and-Spoke Networks 128
How to Configure Next Hop Resolution Protocol 128
Enabling NHRP on an Interface 128
Configuring a GRE Tunnel for Multipoint Operation 129
Configuration Examples for Next Hop Resolution Protocol 131
Physical Network Designs for Logical NBMA Examples 131
Example: GRE Tunnel for Multipoint Operation 133
Additional References for Configuring NHRP 134
Feature History for Next Hop Resolution Protocol 134

CHAPTER 13 **Configuring Network Address Translation** 137
Restrictions For Network Address Translation 137
Information About Network Address Translation 137
Network Address Translation 137
Benefits of Configuring Network Address Translation 138

How Network Address Translation Works 138
Uses of Network Address Translation 139
Network Address Translation Inside and Outside Addresses 139
VRF-Aware Network Address Translation 140
Route Map-based Network Address Translation 140
Limitations of Route Map based Network Address Translation 141
Types of Network Address Translation 142
Using Network Address Translation to Route Packets to the Outside Network (Inside Source Address Translation) 142
Outside Source Address Translation 144
Port Address Translation (PAT) 144
Overlapping Networks 145
Address Only Translation 146
Restrictions for Address Only Translation 147
Limitations of Network Address Translation 147
Performance and Scale Numbers for Network Address Translation 148
Using Application-Level Gateways with Network Address Translation 149
Best Practices for NAT Configuration 149
Configuring Network Address Translation 149
Configuring Static Translation of Inside Source Addresses 150
Configuring Dynamic Translation of Inside Source Addresses 151
Configuring Port Address Translation 153
Configuring Port Address Translation by Overloading of Global Addresses 153
Configuring Port Address Translation by Overloading an Interface 155
Configuring Network Address Translation of External IP Addresses Only 156
Configuring Translation of Overlapping Networks 158
Configuring Address Translation Timeouts 159
Configuring Switch Database Management (SDM) Template 161
Configuring a Static Rule Using a Route Map 162
Configuring a Dynamic Rule Using a Route Map 163
Configuring Rate Limit 165
Configuring Network Address Translation on Layer 3 Port Channel 166
Configuration Examples for Network Address Translation 167
Example: Configuring Static Translation of Inside Source Addresses 167

Example: Configuring Dynamic Translation of Inside Source Addresses 167
Troubleshooting NAT 168
Feature History for Network Address Translation 168

CHAPTER 14 **Configuring Stateful Network Address Translation 64** 171
Restrictions for Configuring Stateful Network Address Translation 64 171
Information About Stateful Network Address Translation 64 172
Stateful Network Address Translation 64 172
Prefix Format for Stateful Network Address Translation 64 173
Well-Known Prefix 173
Performance and Scale Numbers for Stateful Nat64 173
Stateful IPv4-to-IPv6 Packet Flow 173
Stateful IPv6-to-IPv4 Packet Flow 174
VRF-Aware Network Address Translation 64 174
How to Configure Stateful Network Address Translation 64 176
Configuring Static Stateful Network Address Translation 64 176
Configuring Dynamic Stateful Network Address Translation 64 178
Configuring Dynamic Port Address Translation Stateful NAT64 181
Configuring Timeout Functionality 183
Configuring Switch Database Management Template 184
Configuration Examples for Stateful Network Address Translation 64 185
Example: Configuring Static Stateful Network Address Translation 64 185
Example: Configuring Dynamic Stateful Network Address Translation 64 186
Example: Configuring Dynamic Port Address Translation Stateful NAT64 186
Feature History for Configuring Stateful Network Address Translation 64 187

CHAPTER 15 **VRRPv3 Protocol Support** 189
Restrictions for VRRPv3 Protocol Support 189
Information About VRRPv3 Protocol Support 190
VRRPv3 Benefits 190
VRRP Device Priority and Preemption 191
VRRP Advertisements 192
How to Configure VRRPv3 Protocol Support 192
Creating and Customizing a VRRP Group 192

Configuring the Delay Period Before FHRP Client Initialization 194
Configuration Examples for VRRPv3 Protocol Support 195
Example: Enabling VRRPv3 on a Device 195
Example: Creating and Customizing a VRRP Group 195
Example: Configuring the Delay Period Before FHRP Client Initialization 196
Example: VRRP Status, Configuration, and Statistics Details 196
Additional References 197
Feature History for VRRPv3 Protocol Support 197

CHAPTER 16 **Configuring WCCP** 199
Prerequisites for WCCP 199
Restrictions for WCCP 199
Information About WCCP 200
WCCP Overview 201
WCCP Mask Assignment 201
WCCPv2 Configuration 201
WCCPv2 Support for Services Other than HTTP 203
WCCPv2 Support for Multiple Devices 203
WCCPv2 MD5 Security 203
WCCPv2 Web Cache Packet Return 203
WCCPv2 Load Distribution 204
WCCP Bypass Packets 204
WCCP Closed Services and Open Services 204
WCCP Outbound ACL Check 204
WCCP Service Groups 205
WCCP: Check All Services 206
WCCP VRF 206
WCCP Troubleshooting Tips 206
How to Configure WCCP 207
Configuring WCCP 207
Configuring Closed Services 208
Registering a Device to a Multicast Address 210
Using Access Lists for a WCCP Service Group 211
Enabling the WCCP Outbound ACL Check 213

Verifying and Monitoring WCCP Configuration Settings 214
Configuration Examples for WCCP 214
Example: Configuring a General WCCPv2 Session 215
Example: Setting a Password for a Device and Content Engines 215
Example: Configuring a Web Cache Service 215
Example: Running a Reverse Proxy Service 215
Example: Registering a Device to a Multicast Address 216
Example: Using Access Lists 216
Example: WCCP Outbound ACL Check Configuration 216
Example: Verifying WCCP Settings 217
Feature History for WCCP 219

CHAPTER 17 **Configuring Enhanced Object Tracking 221**
Restrictions for Enhanced Object Tracking 221
Information About Enhanced Object Tracking 221
Enhanced Object Tracking Overview 221
Tracking Interface Line-Protocol or IP Routing State 222
Tracked Lists 222
Tracking Other Characteristics 222
IP SLAs Object Tracking 223
Static Route Object Tracking 223
How to Configure Enhanced Object Tracking 223
Configuring Tracking for Line State Protocol or IP Routing State on an Interface 223
Configuring Tracked Lists 224
Configuring a Tracked List with a Weight Threshold 224
Configuring a Tracked List with a Percentage Threshold 226
Configuring HSRP Object Tracking 227
Configuring IP SLAs Object Tracking 230
Configuring Static Route Object Tracking 230
Configuring a Primary Interface for Static Routing 231
Configuring a Primary Interface for DHCP 231
Configuring IP SLAs Monitoring Agent 232
Configuring a Routing Policy and a Default Route 233
Monitoring Enhanced Object Tracking 235

Feature History for Enhanced Object Tracking 235

CHAPTER 18 **Configuring TCP MSS Adjustment** 237

Restrictions for TCP MSS Adjustment 237

Information about TCP MSS Adjustment 237

How to Configure TCP MSS Adjustment 238

Configuring the MSS Value for Transient TCP SYN Packets 238

Configuring the MSS Value for IPv6 Traffic 239

Configuration Examples for TCP MSS Adjustment 239

Example: Configuring the TCP MSS Adjustment 240

Example: Configuring the TCP MSS Adjustment for IPv6 traffic 240

Feature History for TCP MSS Adjustment 240

CHAPTER 19 **Enhanced IPv6 Neighbor Discovery Cache Management** 241

Enhanced IPv6 Neighbor Discovery Cache Management 241

Customizing the Parameters for IPv6 Neighbor Discovery 242

Examples: Customizing Parameters for IPv6 Neighbor Discovery 243

Additional References 243

Feature History for IPv6 Neighbor Discovery 243

CHAPTER 20 **IPv6 Neighbor Discovery Proxy** 245

Prerequisites for IPv6 Neighbor Discovery Proxy 245

Restrictions for IPv6 Neighbor Discovery Proxy 245

Information About IPv6 Neighbor Discovery Proxy 245

How to Configure IPv6 Neighbor Discovery Proxy 246

Configuring IPv6 Routing Proxy in VLAN Configuration Mode 246

Configuring IPv6 Routing Proxy on an Interface 247

Configuring IPv6 DAD Proxy in VLAN Configuration Mode 248

Configuring IPv6 DAD Proxy on an Interface 249

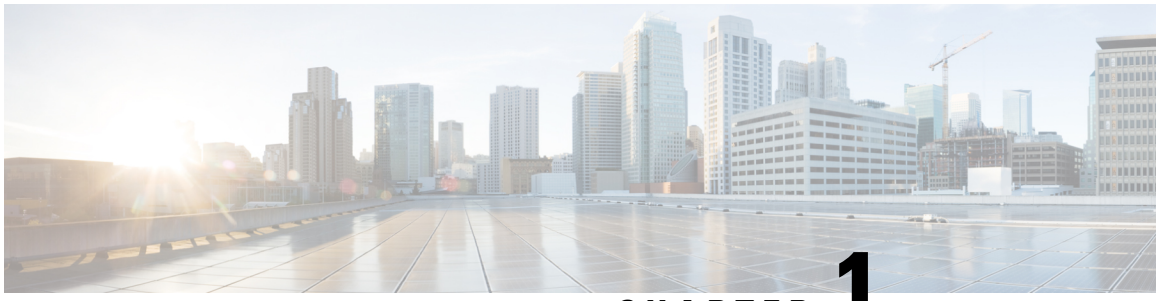
Verifying IPv6 Neighbor Discovery Proxy 251

Configuration Examples For IPv6 Neighbor Discovery Proxy 251

Feature History for IPv6 Neighbor Discovery Proxy 252

CHAPTER 21 **Troubleshooting IP Addressing Services** 253

Overview 253
Support Articles 253
Feedback Request 254
Disclaimer and Caution 254



CHAPTER 1

IP Addressing Services Overview

This section provides information about IP Addressing Services.

- Understanding IPv6, on page 1
- IPv6 Addresses, on page 1
- 128-Bit Wide Unicast Addresses, on page 2
- DNS for IPv6, on page 2
- IPv6 Stateless Autoconfiguration and Duplicate Address Detection, on page 3
- IPv6 Applications, on page 3
- DHCP for IPv6 Address Assignment, on page 3
- HTTP(S) Over IPv6, on page 3

Understanding IPv6

IPv4 users can move to IPv6 and receive services such as end-to-end security, quality of service (QoS), and globally unique addresses. The IPv6 address space reduces the need for private addresses and Network Address Translation (NAT) processing by border routers at network edges.

For information about how Cisco Systems implements IPv6, go to Networking Software (IOS & NX-OS)

For information about IPv6 and other features in this chapter

- See the *Cisco IOS IPv6 Configuration Library*.
- Use the Search field on Cisco.com to locate the Cisco IOS software documentation. For example, if you want information about static routes, you can enter *Implementing Static Routes for IPv6* in the search field to learn about static routes.

IPv6 Addresses

The switch supports only IPv6 unicast addresses. It does not support site-local unicast addresses, or anycast addresses.

The IPv6 128-bit addresses are represented as a series of eight 16-bit hexadecimal fields separated by colons in the format: n:n:n:n:n:n:n:n. This is an example of an IPv6 address:

2031:0000:130F:0000:0000:09C0:080F:130B

For easier implementation, leading zeros in each field are optional. This is the same address without leading zeros:

2031:0:130F:0:0:9C0:80F:130B

You can also use two colons (::) to represent successive hexadecimal fields of zeros, but you can use this short version only once in each address:

2031:0:130F::09C0:080F:130B

For more information about IPv6 address formats, address types, and the IPv6 packet header, see the IPv6 Addressing and Basic Connectivity Configuration Guide of *Cisco IOS IPv6 Configuration Library* on Cisco.com.

- IPv6 Address Formats
- IPv6 Address Type: Multicast
- IPv6 Address Output Display
- Simplified IPv6 Packet Header

128-Bit Wide Unicast Addresses

The switch supports aggregatable global unicast addresses and link-local unicast addresses. It does not support site-local unicast addresses.

- Aggregatable global unicast addresses are IPv6 addresses from the aggregatable global unicast prefix. The address structure enables strict aggregation of routing prefixes and limits the number of routing table entries in the global routing table. These addresses are used on links that are aggregated through organizations and eventually to the Internet service provider.

  These addresses are defined by a global routing prefix, a subnet ID, and an interface ID. Current global unicast address allocation uses the range of addresses that start with binary value 001 (2000::/3). Addresses with a prefix of 2000::/3(001) through E000::/3(111) must have 64-bit interface identifiers in the extended unique identifier (EUI)-64 format.

- Link local unicast addresses can be automatically configured on any interface by using the link-local prefix FE80::/10(1111 1110 10) and the interface identifier in the modified EUI format. Link-local addresses are used in the neighbor discovery protocol (NDP) and the stateless autoconfiguration process. Nodes on a local link use link-local addresses and do not require globally unique addresses to communicate. IPv6 routers do not forward packets with link-local source or destination addresses to other links.

For more information, see the section about IPv6 unicast addresses in the “Implementing IPv6 Addressing and Basic Connectivity” chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

DNS for IPv6

IPv6 supports Domain Name System (DNS) record types in the DNS name-to-address and address-to-name lookup processes. The DNS AAAA resource record types support IPv6 addresses and are equivalent to an A address record in IPv4. The switch supports DNS resolution for IPv4 and IPv6.

IPv6 Stateless Autoconfiguration and Duplicate Address Detection

The switch uses stateless autoconfiguration to manage link, subnet, and site addressing changes, such as management of host and mobile IP addresses. A host autonomously configures its own link-local address, and booting nodes send router solicitations to request router advertisements for configuring interfaces.

For more information about autoconfiguration and duplicate address detection, see the "Implementing IPv6 Addressing and Basic Connectivity" chapter of *Cisco IOS IPv6 Configuration Library* on Cisco.com.

IPv6 Applications

The switch has IPv6 support for these applications:

- Ping, traceroute, Telnet, and TFTP
- Secure Shell (SSH) over an IPv6 transport
- HTTP server access over IPv6 transport
- DNS resolver for AAAA over IPv4 transport
- Cisco Discovery Protocol (CDP) support for IPv6 addresses

For more information about managing these applications, see the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

DHCP for IPv6 Address Assignment

DHCPv6 enables DHCP servers to pass configuration parameters, such as IPv6 network addresses, to IPv6 clients. The address assignment feature manages non-duplicate address assignment in the correct prefix based on the network where the host is connected. Assigned addresses can be from one or multiple prefix pools. Additional options, such as default domain and DNS name-server address, can be passed back to the client. Address pools can be assigned for use on a specific interface, on multiple interfaces, or the server can automatically find the appropriate pool.

For configuring DHCP for IPv6, see the *Configuring DHCP for IPv6 Address Assignment* section.

For more information about configuring the DHCPv6 client, server, or relay agent functions, see the *Cisco IOS IPv6 Configuration Library* on Cisco.com.

HTTP(S) Over IPv6

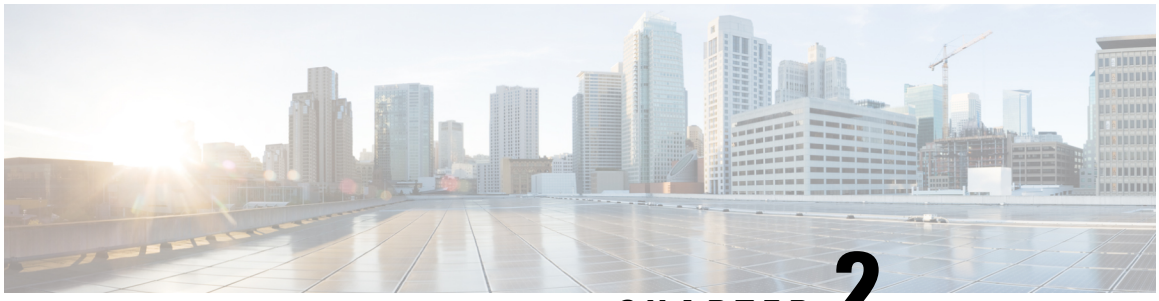
The HTTP client sends requests to both IPv4 and IPv6 HTTP servers, which respond to requests from both IPv4 and IPv6 HTTP clients. URLs with literal IPv6 addresses must be specified in hexadecimal using 16-bit values between colons.

The accept socket call chooses an IPv4 or IPv6 address family. The accept socket is either an IPv4 or IPv6 socket. The listening socket continues to listen for both IPv4 and IPv6 signals that indicate a connection. The IPv6 listening socket is bound to an IPv6 wildcard address.

The underlying TCP/IP stack supports a dual-stack environment. HTTP relies on the TCP/IP stack and the sockets for processing network-layer interactions.

Basic network connectivity (**ping**) must exist between the client and the server hosts before HTTP connections can be made.

For more information, see the "Managing Cisco IOS Applications over IPv6" chapter in the *Cisco IOS IPv6 Configuration Library* on Cisco.com.



CHAPTER 2

IPv6 Client IP Address Learning

- Prerequisites for IPv6 Client Address Learning, on page 5
- Information About IPv6 Client Address Learning, on page 5
- How to Configure IPv6 Client Address Learning, on page 9
- Verifying IPv6 Address Learning Configuration, on page 22
- Additional References, on page 23
- Feature History for IPv6 Client Address Learning, on page 23

Prerequisites for IPv6 Client Address Learning

Before configuring IPv6 client address learning, configure the clients to support IPv6.

Information About IPv6 Client Address Learning

Client Address Learning is configured on device to learn the client's IPv4 and IPv6 address and clients transition state maintained by the device on an association, re-association, de-authentication and timeout.

There are three ways for IPv6 client to acquire IPv6 addresses:

- Stateless Address Auto-Configuration (SLACC)
- Stateful DHCPv6
- Static Configuration

For all of these methods, the IPv6 client always sends neighbor solicitation DAD (Duplicate Address Detection) request to ensure there is no duplicate IP address on the network. The device snoops the client's Neighbor Discovery Protocol (NDP) and DHCPv6 packets to learn about its client IP addresses.

When a duplicate IPv6 address is configured, DAD detects the duplicate address, and advertises it in the Router Advertisement (RA). The duplicate address can be manually removed from the system, so that it is not displayed in the connected address and not advertised in the RA prefix.

SLAAC Address Assignment

The most common method for IPv6 client address assignment is Stateless Address Auto-Configuration (SLAAC). SLAAC provides simple plug-and-play connectivity where clients self-assign an address based on the IPv6 prefix. This process is achieved

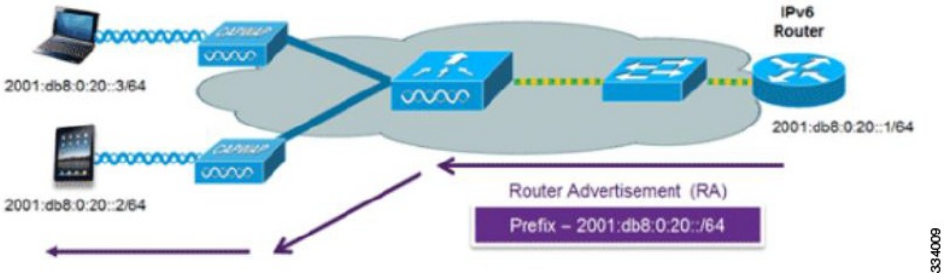
Stateless Address Auto-Configuration (SLAAC) is configured as follows:

- Host sends a router solicitation message.
- Hosts waits for a Router Advertisement message.
- Hosts take the first 64 bits of the IPv6 prefix from the Router Advertisement message and combines it with the 64 bit EUI-64 address (in the case of ethernet, this is created from the MAC Address) to create a global unicast message. The host also uses the source IP address, in the IP header, of the Router Advertisement message, as its default gateway.
- Duplicate Address Detection is performed by IPv6 clients in order to ensure that random addresses that are picked do not collide with other clients.
- The choice of algorithm is up to the client and is often configurable.

The last 64 bits of the IP v6 address can be learned based on the following 2 algorithms:

- EUI-64 which is based on the MAC address of the interface, or
- Private addresses that are randomly generated.

Figure 1: SLAAC Address Assignment

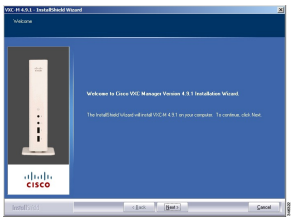


The following Cisco IOS configuration commands from a Cisco-capable IPv6 router are used to enable SLAAC addressing and router advertisements:

```
ipv6 unicast-routing
interface Vlan20
description IPv6-SLAAC
ip address 192.168.20.1 255.255.255.0
ipv6 address FE80:DB8:0:20::1 linklocal
ipv6 address 2001:DB8:0:20::1/64
ipv6 enable
end
```

Stateful DHCPv6 Address Assignment

Figure 2: Stateful DHCPv6 Address Assignment



The use of DHCPv6 is not required for IPv6 client connectivity if SLAAC is already deployed. There are two modes of operation for DHCPv6 called Stateless and Stateful.

The DHCPv6 Stateless mode is used to provide clients with additional network information that is not available in the router advertisement, but not an IPv6 address as this is already provided by SLAAC. This information can include the DNS domain name, DNS server(s), and other DHCP vendor-specific options. This interface configuration is for a Cisco IOS IPv6 router implementing stateless DHCPv6 with SLAAC enabled:

```
ipv6 unicast-routing
ipv6 dhcp pool IPV6_DHCPPOOL
address prefix 2001:db8:5:10::/64
domain-name cisco.com
dns-server 2001:db8:6:6::1
interface Vlan20
description IPv6-DHCP-Stateless
ip address 192.168.20.1 255.255.255.0
ipv6 nd other-config-flag
ipv6 dhcp server IPV6_DHCPPOOL
ipv6 address 2001:DB8:0:20::1/64
end
```

The DHCPv6 Stateful option, also known as managed mode, operates similarly to DHCPv4 in that it assigns unique addresses to each client instead of the client generating the last 64 bits of the address as in SLAAC. This interface configuration is for a Cisco IOS IPv6 router implementing stateful DHCPv6 on a local device:

```
ipv6 unicast-routing
ipv6 dhcp pool IPV6_DHCPPOOL
address prefix 2001:db8:5:10::/64
domain-name cisco.com
dns-server 2001:db8:6:6::1
interface Vlan20
description IPv6-DHCP-Stateful
ip address 192.168.20.1 255.255.255.0
ipv6 address 2001:DB8:0:20::1/64
ipv6 nd prefix 2001:DB8:0:20::/64 no-advertise
ipv6 nd managed-config-flag
ipv6 nd other-config-flag
ipv6 dhcp server IPV6_DHCPPOOL
end
```

This interface configuration is for a Cisco IOS IPv6 router implementing stateful DHCPv6 on an external DHCP server:

```
ipv6 unicast-routing
domain-name cisco.com
dns-server 2001:db8:6:6::1
interface Vlan20
description IPv6-DHCP-Stateful
ip address 192.168.20.1 255.255.255.0
ipv6 address 2001:DB8:0:20::1/64
```

```
ipv6 nd prefix 2001:DB8:0:20::/64 no-advertise
ipv6 nd managed-config-flag
ipv6 nd other-config-flag
ipv6 dhcp relay destination 2001:DB8:0:20::2
end
```

Static IP Address Assignment

Statically configured address on a client.

Router Solicitation

A Router Solicitation message is issued by a host to facilitate local routers to transmit Router Advertisement from which it can obtain information about local routing or perform Stateless Auto-configuration. Router Advertisements are transmitted periodically and the host prompts with an immediate Router Advertisement using a Router Solicitation such as - when it boots or following a restart operation.

Router Advertisement

A Router Advertisement message is issued periodically by a router or in response to a Router Solicitation message from a host. The information contained in these messages is used by hosts to perform Stateless Auto-configuration and to modify its routing table.

Neighbor Discovery

IPv6 Neighbor Discovery is a set of messages and processes that determine relationships between neighboring nodes. Neighbor Discovery replaces ARP, ICMP Router Discovery, and ICMP Redirect used in IPv4.

IPv6 Neighbor Discovery inspection analyzes neighbor discovery messages in order to build a trusted binding table database, and IPv6 neighbor discovery packets that do not comply are dropped. The neighbor binding table in the switch tracks each IPv6 address and its associated MAC address. Clients are expired from the table according to Neighbor Binding timers.

Neighbor Discovery Suppression

The IPv6 addresses of clients are cached by the device. When the device receives an NS multicast looking for an IPv6 address, and if the target address is known to the device and belongs to one of its clients, the device will reply with an NA message on behalf of the client. The result of this process generates the equivalent of the Address Resolution Protocol (ARP) table of IPv4 but is more efficient - uses generally fewer messages.

Note The device acts like proxy and respond with NA, only when the **ipv6 nd suppress** command is configured

If the device does not have the IPv6 address of a client, the device will not respond with NA and forward the NS packet . To resolve this, an NS Multicast Forwarding knob is provided. If this knob is enabled, the device gets the NS packet for the IPv6 address that it does not have (cache miss) and forwards it . This packet reaches the intended client and the client replies with NA.

This cache miss scenario occurs rarely, and only very few clients which do not implement complete IPv6 stack may not advertise their IPv6 address during NDP.

RA Guard

IPv6 clients configure IPv6 addresses and populate their router tables based on IPv6 router advertisement (RA) packets. The RA guard feature is similar to the RA guard feature of wired networks. RA guard increases the security of the IPv6 network by dropping the unwanted or rogue RA packets that come from clients. If this feature is not configured, malicious IPv6 clients announce themselves as the router for the network often with high priority, which would take higher precedence over legitimate IPv6 routers.

RA-Guard also examines the incoming RA's and decides whether to switch or block them based solely on information found in the message or in the switch configuration. The information available in the frames received is useful for RA validation:

- Port on which the frame is received
- IPv6 source address
- Prefix list

The following configuration information created on the switch is available to RA-Guard to validate against the information found in the received RA frame:

- Trusted/Untrusted ports for receiving RA-guard messages
- Trusted/Untrusted IPv6 source addresses of RA-sender
- Trusted/Untrusted Prefix list and Prefix ranges
- Router Preference

RA guard is applied on the device. You can configure the device to drop RA messages on the device. All IPv6 RA messages are dropped, which protects other clients and upstream wired network from malicious IPv6 clients.

```
//Create a policy for RA Guard//
ipv6 nd raguard policy raguard-router
trusted-port
device-role router

//Applying the RA Guard Policy on port/interface//
interface tengigabitethernet1/0/1 (Katana)
interface gigabitethernet1/0/1 (Edison)

ipv6 nd raguard attach-policy raguard-router
```

How to Configure IPv6 Client Address Learning

The following sections provide configuration information about IPv6 client address learning.

Configuring IPv6 Unicast

IPv6 unicasting must always be enabled on the switch. IPv6 unicast routing is disabled.

To configure IPv6 unicast, perform this procedure:

Before you begin

To enable the forwarding of IPv6 unicast datagrams, use the **ipv6 unicast-routing** command in global configuration mode. To disable the forwarding of IPv6 unicast datagrams, use the **no** form of this command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast routing** **Example:** `Device(config)# ipv6 unicast routing`	enable the forwarding of IPv6 unicast datagrams

Configuring RA Guard Policy

Configure RA Guard policy on the device to add IPv6 client addresses and populate the router table based on IPv6 router advertisement packets.

To configuring RA guard policy, perform this procedure:

Procedure

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

	Command or Action	Purpose
Step 3	**ipv6 nd raguard policy raguard-router** **Example:** `Device(config)# ipv6 nd raguard policy raguard-router`	Defines the RA guard policy name and enters RA guard policy configuration mode.
Step 4	**trustedport** **Example:** `Device(config-ra-guard)# trustedport`	(Optional) Specifies that this policy is being applied to trusted ports.
Step 5	**device-role router** **Example:** `Device(config-ra-guard)# device-role router`	Specifies the role of the device attached to the port.
Step 6	**exit** **Example:** `Device(config-ra-guard)# exit`	Exits RA guard policy configuration mode and returns to global configuration mode.

Applying RA Guard Policy

Applying the RA Guard policy on the device will block all the untrusted RA's.

To apply RA guard policy, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface tengigabitethernet 1/0/1** **Example:** `Device(config)# interface tengigabitethernet 1/0/1`	Specifies an interface type and number, and places the device in interface configuration mode.

	Command or Action	Purpose
Step 4	**ipv6 nd raguard attach-policy raguard-router** **Example:** `Device(config-if)# ipv6 nd raguard attach-policy raguard-router`	Applies the IPv6 RA Guard feature to a specified interface.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.

Configuring IPv6 Snooping

Note We recommend that you configure SISF-based device tracking configurations instead of IPv6 snooping legacy configuration. For more information, refer to the *Configuring SISF-Based Device Tracking* section in the *Security Configuration Guide*.

IPv6 snooping must always be enabled on the switch.

To configuring IPv6 snooping, perform this procedure:

Before you begin

Enable IPv6 on the client machine.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan configuration 1** **Example:** `Device(config)# vlan configuration 1`	Enters VLAN configuration mode.

	Command or Action	Purpose
Step 4	**ipv6 snooping** **Example:** `Device(config-vlan)# ipv6 snooping`	Enables IPv6 snooping on the Vlan.
Step 5	**ipv6 nd suppress** **Example:** `Device(config-vlan-config)# ipv6 nd suppress`	Enables the IPv6 ND suppress on the Vlan.
Step 6	**exit** **Example:** `Device(config-vlan-config)# exit`	Saves the configuration and comes out of the Vlan configuration mode.

Configuring IPv6 ND Suppress Policy

The IPv6 neighbor discovery (ND) multicast suppress feature stops as many ND multicast neighbor solicit (NS) messages as possible by dropping them (and responding to solicitations on behalf of the targets) or converting them into unicast traffic. This feature runs on a layer 2 switch and is used to reduce the amount of control traffic necessary for proper link operations.

When an address is inserted into the binding table, an address resolution request sent to a multicast address is intercepted, and the device either responds on behalf of the address owner or, at layer 2, converts the request into a unicast message and forwards it to its destination.

To configure IPv6 ND suppress policy, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 nd suppress policy** *policy_name* **Example:** `Device(config)# ipv6 nd suppress policy policy1`	Defines the ND suppress policy name and enters ND suppress policy configuration mode.

Configuring IPv6 Snooping on VLAN/PortChannel

Neighbor Discover (ND) suppress can be enabled or disabled on either the VLAN or a switchport.

To configure IPv6 snooping on VLAN/PortChannel, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan config901** **Example:** `Device(config)# vlan config901`	Creates a VLAN and enter the VLAN configuration mode
Step 4	**ipv6 nd suppress** **Example:** `Device(config-vlan)# ipv6 nd suppress`	Applies the IPv6 nd suppress on VLAN.
Step 5	**end** **Example:** `Device(config-vlan)# end`	Exits vlan configuration mode and enters the global configuration mode.
Step 6	**interface gi1/0/1** **Example:** `Device(config)# interface gi1/0/1`	Creates a gigabitethernet port interface.
Step 7	**ipv6 nd suppress** **Example:** `Device(config-vlan)# ipv6 nd suppress`	Applies the IPv6 nd suppress on the interface.
Step 8	**end** **Example:** `Device(config-vlan)# end`	Exits vlan configuration mode and enters the global configuration mode.

Configuring IPv6 on Switch Interface

Follow the procedure given below to configure IPv6 on an interface:

Before you begin

Enable IPv6 on the client and IPv6 support on the wired infrastructure.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface vlan 1** **Example:** `Device(config)# interface vlan 1`	Creates a interface and enters interface configuration mode.

Step 4	**ip address fe80::1 link-local**
	Example:
	`Device(config-if)# ip address 198.51.100.1 255.255.255.0`
	`Device(config-if)# ipv6 address fe80::1 link-local`
	`Device(config-if)# ipv6 address 2001:DB8:0:1:FFFF:1234::5/64`
	`Device(config-if)# ipv6 address 2001:DB8:0:0:E000::F/64`
		Configures IPv6 address on the interface using the link-local option.

Step 5	**ipv6 enable** **Example:** `Device(config)# ipv6 enable`	(Optional) Enables IPv6 on the interface.
Step 6	**end** **Example:** `Device(config)# end`	Exits from the interface mode.

Configuring DHCP Pool on Switch Interface

Follow the procedure given below to configure DHCP Pool on an interface:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** `Vlan21` **Example:** `Device(config)# ipv6 dhcp pool vlan1`	Enters the configuration mode and configures the IPv6 DHCP pool on the Vlan.
Step 4	**address prefix** `2001:DB8:0:1:FFFF:1234::/64` **lifetime** `300 10` **Example:** `Device(config-dhcpv6)# address prefix 2001:DB8:0:1:FFFF:1234::/64 lifetime 300 10`	Enters the configuration-dhcp mode and configures the address pool and its lifetime on a Vlan.
Step 5	**dns-server** `2001:100:0:1::1` **Example:** `Device(config-dhcpv6)# dns-server 2001:20:21::1`	Configures the DNS servers for the DHCP pool.
Step 6	**domain-name** `example.com` **Example:** `Device(config-dhcpv6)# domain-name example.com`	Configures the domain name to complete unqualified host names.
Step 7	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

Configuring Stateless Auto Address Configuration Without DHCP

Follow the procedure given below to configure stateless auto address configuration without DHCP:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface vlan 1** **Example:** `Device(config)# interface vlan 1`	Creates a interface and enters interface configuration mode.

Step 4	**ip address fe80::1 link-local**
	Example:
	`Device(config-if)# ip address 198.51.100.1 255.255.255.0`
	`Device(config-if)# ipv6 address fe80::1 link-local`
	`Device(config-if)# ipv6 address 2001:DB8:0:1:FFFF:1234::5/64`
	`Device(config-if)# ipv6 address 2001:DB8:0:0:E000::F/64`
		Configures IPv6 address on the interface using the link-local option.

Step 5	**ipv6 enable** **Example:** `Device(config)# ipv6 enable`	(Optional) Enables IPv6 on the interface.

Step 6	**no ipv6 nd managed-config-flag**
	Example:
	`Device(config)# interface vlan 1`
	`Device(config-if)# no ipv6 nd managed-config-flag`
		Ensures the attached hosts do not use stateful autoconfiguration to obtain addresses.

Step 7	**no ipv6 nd other-config-flag** **Example:** `Device(config-if)# no ipv6 nd other-config-flag`	Ensures the attached hosts do not use stateful autoconfiguration to obtain non-address options from DHCP (domain etc).
Step 8	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

Configuring Stateless Auto Address Configuration With DHCP

Follow the procedure given below to configure stateless auto address configuration with DHCP:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface vlan 1** **Example:** `Device(config)# interface vlan 1`	Creates a interface and enters interface configuration mode.

Step 4	**ip address fe80::1 link-local**
	Example:
	`Device(config-if)# ip address 198.51.100.1 255.255.255.0`
	`Device(config-if)# ipv6 address fe80::1 link-local`
	`Device(config-if)# ipv6 address 2001:DB8:0:1:FFFF:1234::5/64`
	`Device(config-if)# ipv6 address 2001:DB8:0:0:E000::F/64`
		Configures IPv6 address on the interface using the link-local option.

Step 5	**ipv6 enable** **Example:** `Device(config)# ipv6 enable`	(Optional) Enables IPv6 on the interface.

Step 6	**no ipv6 nd managed-config-flag**
	Example:
	`Device(config)# interface vlan 1`
	`Device(config-if)# no ipv6 nd managed-config-flag`
		Ensures the attached hosts do not use stateful autoconfiguration to obtain addresses.

Step 7	**ipv6 nd other-config-flag** **Example:** `Device(config-if)# no ipv6 nd other-config-flag`	Ensures the attached hosts do not use stateful autoconfiguration to obtain non-address options from DHCP (domain etc).

	Command or Action	Purpose
Step 8	**end** **Example:** `Device(config)# end`	Exits from the interface mode.

Configuring Stateful DHCP Locally

This interface configuration is for a Cisco IOS IPv6 router implementing stateful DHCPv6 on a local device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Configures IPv6 for unicasting.
Step 4	**ipv6 dhcp pool IPv6_DHCPPOOL** **Example:** `Device(config)# ipv6 dhcp pool IPv6_DHCPPOOL`	Enters the configuration mode and configures the IPv6 DHCP pool on the VLAN.
Step 5	**address prefix 2001:DB8:0:1:FFFF:1234::/64** **Example:** `Device(config-dhcpv6)# address prefix 2001:DB8:0:1:FFFF:1234::/64`	Specifies the address range to provide in the pool.
Step 6	**dns-server 2001:100:0:1::1** **Example:** `Device(config-dhcpv6)# dns-server 2001:100:0:1::1`	Provides the DNS server option to DHCP clients.
Step 7	**domain-name example.com** **Example:**	Provides the domain name option to DHCP clients.

	Command or Action	Purpose
	`Device(config-dhcpv6)# domain-name example.com`	
Step 8	**exit** **Example:** `Device(config-dhcpv6)# exit`	Returns to the previous mode.
Step 9	**interface vlan1** **Example:** `Device(config)# interface vlan 1`	Enters the interface mode to configure the stateful DHCP.
Step 10	**description IPv6-DHCP-Stateful** **Example:** `Device(config-if)# description IPv6-DHCP-Stateful`	Enter description for the stateful IPv6 DHCP.
Step 11	**ipv6 address 2001:DB8:0:20::1/64** **Example:** `Device(config-if)# ipv6 address 2001:DB8:0:20::1/64`	Enters the IPv6 address for the stateful IPv6 DHCP.
Step 12	**ip address 192.168.20.1 255.255.255.0** **Example:** `Device(config-if)# ip address 192.168.20.1 255.255.255.0`	Enters the IPv6 address for the stateful IPv6 DHCP.
Step 13	**ipv6 nd prefix 2001:db8::/64 no-advertise** **Example:** `Device(config-if)# ipv6 nd prefix 2001:db8::/64 no-advertise`	Configures the IPv6 routing prefix advertisement that must not be advertised.
Step 14	**ipv6 nd managed-config-flag** **Example:** `Device(config-if)# ipv6 nd managed-config-flag`	Configures IPv6 interfaces neighbor discovery to allow the hosts to uses DHCP for address configuration.
Step 15	**ipv6 nd other-config-flag** **Example:** `Device(config-if)# ipv6 nd other-config-flag`	Configures IPv6 interfaces neighbor discovery to allow the hosts to uses DHCP for non-address configuration.
Step 16	**ipv6 dhcp server IPv6_DHCPPOOL** **Example:** `Device(config-if)# ipv6 dhcp server IPv6_DHCPPOOL`	Configures the DHCP server on the interface.

Configuring Stateful DHCP Externally

This interface configuration is for a Cisco IOS IPv6 router implementing stateful DHCPv6 on an external DHCP server.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Configures the IPv6 for unicasting.
Step 4	**dns-server** `2001:100:0:1::1` **Example:** `Device(config-dhcpv6)# dns-server 2001:100:0:1::1`	Provides the DNS server option to DHCP clients.
Step 5	**domain-name** `example.com` **Example:** `Device(config-dhcpv6)# domain-name example.com`	Provides the domain name option to DHCP clients.
Step 6	**exit** **Example:** `Device(config-dhcpv6)# exit`	Returns to the previous mode.
Step 7	**interface** `vlan 1` **Example:** `Device(config)# interface vlan 1`	Enters the interface mode to configure the stateful DHCP.
Step 8	**description IPv6-DHCP-Stateful** **Example:** `Device(config-if)# description IPv6-DHCP-Stateful`	Enter description for the stateful IPv6 DHCP.
Step 9	**ipv6 address** `2001:DB8:0:20::1/64` **Example:**	Enters the IPv6 address for the stateful IPv6 DHCP.

	Command or Action	Purpose
	`Device(config-if)# ipv6 address 2001:DB8:0:20::1/64`	
Step 10	**ip address 192.168.20.1 255.255.255.0** **Example:** `Device(config-if)# ip address 192.168.20.1 255.255.255.0`	Enters the IPv6 address for the stateful IPv6 DHCP.
Step 11	**ipv6 nd prefix 2001:db8::/64 no-advertise** **Example:** `Device(config-if)# ipv6 nd prefix 2001:db8::/64 no-advertise`	Configures the IPv6 routing prefix advertisement that must not be advertised.
Step 12	**ipv6 nd managed-config-flag** **Example:** `Device(config-if)# ipv6 nd managed-config-flag`	Configures IPv6 interfaces neighbor discovery to allow the hosts to uses DHCP for address configuration.
Step 13	**ipv6 nd other-config-flag** **Example:** `Device(config-if)# ipv6 nd other-config-flag`	Configures IPv6 interfaces neighbor discovery to allow the hosts to uses DHCP for non-address configuration.
Step 14	**ipv6 dhcp relaydestination 2001:DB8:0:20::2** **Example:** `Device(config-if)# ipv6 dhcp relay destination 2001:DB8:0:20::2`	Configures the DHCP server on the interface.

Verifying IPv6 Address Learning Configuration

This example displays the output of the **show ipv6 dhcp pool** command. This command displays the IPv6 service configuration on the device. The vlan 21 configured pool detail displays 6 clients that are currently using addresses from the pool.

Procedure

	Command or Action	Purpose

Step 1	**show ipv6 dhcp pool**
	Example:
	`Device show ipv6 dhcp pool`
	`DHCPv6 pool: vlan21`
	`Address allocation prefix:`
		Displays the IPv6 service configuration on the device.

	Command or Action	Purpose

	`2001:DB8:0:1:FFFF:1234::/64 valid 86400 preferred 86400 (6 in use, 0 conflicts)`
	`DNS server: 2001:100:0:1::1`
	`Domain name: example.com`
	`Active clients: 6`	

Additional References

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9600 Series Switches)*

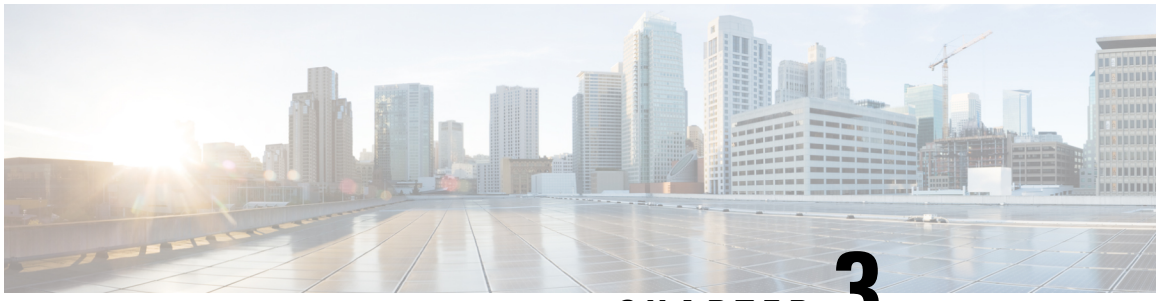
Feature History for IPv6 Client Address Learning

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 Gibraltar 16.11.1	IPv6 Client Address Learning Functionality	Client Address Learning is configured on device to learn the client's IPv4 and IPv6 address and clients transition state maintained by the device on an association, re-association, de-authentication and timeout.
Cisco IOS XE Cupertino 17.7.1	IPv6 Client Address Learning Functionality	This feature was implemented on the Cisco Catalyst 9600 Series Supervisor 2 Module (C9600X-SUP-2).

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/



CHAPTER 3

Configuring DHCP

This section provides information about configuring DHCP.

- Prerequisites for Configuring DHCP, on page 25
- Restrictions for Configuring DHCP, on page 26
- Information About DHCP, on page 26
- How to Configure DHCP, on page 34
- Feature History for DHCP, on page 45

Prerequisites for Configuring DHCP

The following prerequisites apply to DHCP Snooping and Option 82:

- You must globally enable DHCP snooping on the switch.
- Before globally enabling DHCP snooping on the switch, make sure that the devices acting as the DHCP server and the DHCP relay agent are configured and enabled.
- If you want the switch to respond to DHCP requests, it must be configured as a DHCP server.
- Before configuring the DHCP snooping information option on your switch, be sure to configure the device that is acting as the DHCP server. You must specify the IP addresses that the DHCP server can assign or exclude, or you must configure DHCP options for these devices.
- For DHCP snooping to function properly, all DHCP servers must be connected to the switch through trusted interfaces, as untrusted DHCP messages will be forwarded only to trusted interfaces. In a service-provider network, a trusted interface is connected to a port on a device in the same network.
- You must configure the switch to use the Cisco IOS DHCP server binding database to use it for DHCP snooping.
- To use the DHCP snooping option of accepting packets on untrusted inputs, the switch must be an aggregation switch that receives packets with option-82 information from an edge switch.
- The following prerequisites apply to DHCP snooping binding database configuration:
  - You must configure a destination on the DHCP snooping binding database to use the switch for DHCP snooping.
  - Because both NVRAM and the flash memory have limited storage capacity, we recommend that you store the binding file on a TFTP server.

- For network-based URLs (such as TFTP and FTP), you must create an empty file at the configured URL before the switch can write bindings to the binding file at that URL. See the documentation for your TFTP server to determine whether you must first create an empty file on the server; some TFTP servers cannot be configured this way.
- To ensure that the lease time in the database is accurate, we recommend that you enable and configure Network Time Protocol (NTP).
- If NTP is configured, the switch writes binding changes to the binding file only when the switch system clock is synchronized with NTP.

- Before configuring the DHCP relay agent on your switch, make sure to configure the device that is acting as the DHCP server. You must specify the IP addresses that the DHCP server can assign or exclude, configure DHCP options for devices, or set up the DHCP database agent.
- If you want the switch to relay DHCP packets, the IP address of the DHCP server must be configured on the switch virtual interface (SVI) of the DHCP client.
- If a switch port is connected to a DHCP server, configure a port as trusted by entering the **ip dhcp snooping trust interface** configuration command.
- If a switch port is connected to a DHCP client, configure a port as untrusted by entering the **no ip dhcp snooping trust** interface configuration command.

Restrictions for Configuring DHCP

We recommend that you do not use transmit (TX) Remote or Encapsulated Remote Switched Port Analyzer (RSPAN or ERSPAN) on VLAN ports which support DHCP Snooping or DHCP Relay Agent. If TX RSPAN or ERSPAN is required, avoid using VLAN ports that are in the forwarding path for DHCP packets.

Information About DHCP

DHCP Server

The DHCP server assigns IP addresses from specified address pools on a switch or router to DHCP clients and manages them. If the DHCP server cannot give the DHCP client the requested configuration parameters from its database, it forwards the request to one or more secondary DHCP servers defined by the network administrator. The switch can act as a DHCP server. If the DHCP server provides the client with the requested configuration, it will not forward the message to the other server.

DHCP Relay Agent

A DHCP relay agent is a Layer 3 device that forwards DHCP packets between clients and servers. Relay agents forward requests and replies between clients and servers when they are not on the same physical subnet. Relay agent forwarding is different from the normal Layer 2 forwarding, in which IP datagrams are switched transparently between networks. Relay agents receive DHCP messages and generate new DHCP messages to send on output interfaces.

DHCP Snooping

DHCP snooping is a DHCP security feature that provides network security by filtering untrusted DHCP messages and by building and maintaining a DHCP snooping binding database, also referred to as a DHCP snooping binding table.

DHCP snooping acts like a firewall between untrusted hosts and DHCP servers. You use DHCP snooping to differentiate between untrusted interfaces connected to the end user and trusted interfaces connected to the DHCP server or another switch.

Note For DHCP snooping to function properly, all DHCP servers must be connected to the switch through trusted interfaces, as untrusted DHCP messages will be forwarded only to trusted interfaces.

An untrusted DHCP message is a message that is received through an untrusted interface. By default, the switch considers all interfaces untrusted. So, the switch must be configured to trust some interfaces to use DHCP Snooping. When you use DHCP snooping in a service-provider environment, an untrusted message is sent from a device that is not in the service-provider network, such as a customer's switch. Messages from unknown devices are untrusted because they can be sources of traffic attacks.

The DHCP snooping binding database has the MAC address, the IP address, the lease time, the binding type, the VLAN number, and the interface information that corresponds to the local untrusted interfaces of a switch. It does not have information regarding hosts interconnected with a trusted interface.

In a service-provider network, an example of an interface you might configure as trusted is one connected to a port on a device in the same network. An example of an untrusted interface is one that is connected to an untrusted interface in the network or to an interface on a device that is not in the network.

When a switch receives a packet on an untrusted interface and the interface belongs to a VLAN in which DHCP snooping is enabled, the switch compares the source MAC address and the DHCP client hardware address. If the addresses match (the default), the switch forwards the packet. If the addresses do not match, the switch drops the packet.

The switch drops a DHCP packet when one of these situations occurs:

- A packet from a DHCP server, such as a DHCPOFFER, DHCPACK, DHCPNAK, or DHCPLEASEQUERY packet, is received from outside the network or firewall.
- A packet is received on an untrusted interface, and the source MAC address and the DHCP client hardware address do not match.
- The switch receives a DHCPRELEASE or DHCPDECLINE broadcast message that has a MAC address in the DHCP snooping binding database, but the interface information in the binding database does not match the interface on which the message was received.
- A DHCP relay agent forwards a DHCP packet that includes a relay-agent IP address that is not 0.0.0.0, or the relay agent forwards a packet that includes option-82 information to an untrusted port.
- The maximum snooping queue size of 1000 is exceeded when DHCP snooping is enabled.

If the switch is an aggregation switch supporting DHCP snooping and is connected to an edge switch that is inserting DHCP option-82 information, the switch drops packets with option-82 information when packets are received on an untrusted interface. If DHCP snooping is enabled and packets are received on a trusted port, the aggregation switch does not learn the DHCP snooping bindings for connected devices and cannot build a complete DHCP snooping binding database.

When an aggregation switch can be connected to an edge switch through an untrusted interface and you enter the **ip dhcp snooping information option allow-untrusted** global configuration command, the aggregation switch accepts packets with option-82 information from the edge switch. The aggregation switch learns the bindings for hosts connected through an untrusted switch interface. The DHCP security features, such as dynamic ARP inspection or IP source guard, can still be enabled on the aggregation switch while the switch receives packets with option-82 information on untrusted input interfaces to which hosts are connected. The port on the edge switch that connects to the aggregation switch must be configured as a trusted interface.

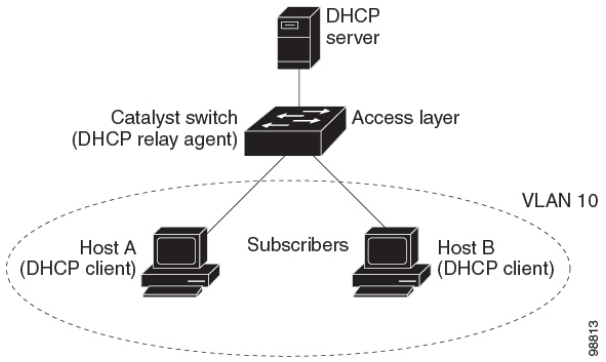
Option-82 Data Insertion

In residential, metropolitan Ethernet-access environments, DHCP can centrally manage the IP address assignments for a large number of subscribers. When the DHCP option-82 feature is enabled on the switch, a subscriber device is identified by the switch port through which it connects to the network (in addition to its MAC address). Multiple hosts on the subscriber LAN can be connected to the same port on the access switch and are uniquely identified.

Note The DHCP option-82 feature is supported only when DHCP snooping is globally enabled on the VLANs to which subscriber devices using option-82 are assigned.

The following illustration shows a metropolitan Ethernet network in which a centralized DHCP server assigns IP addresses to subscribers connected to the switch at the access layer. Because the DHCP clients and their associated DHCP server do not reside on the same IP network or subnet, a DHCP relay agent (the Catalyst switch) is configured with a helper address to enable broadcast forwarding and to transfer DHCP messages between the clients and the server.

Figure 3: DHCP Relay Agent in a Metropolitan Ethernet Network



When you enable the DHCP snooping information option 82 on the switch, the following sequence of events occurs:

- The host (DHCP client) generates a DHCP request and broadcasts it on the network.
- When the switch receives the DHCP request, it adds the option-82 information in the packet. By default, the remote-ID suboption is the switch MAC address, and the circuit-ID suboption is the port identifier, **vlan-mod-port**, from which the packet is received. You can configure the remote ID and circuit ID.
- If the IP address of the relay agent is configured, the switch adds this IP address in the DHCP packet.
- The switch forwards the DHCP request that includes the option-82 field to the DHCP server.

- The DHCP server receives the packet. If the server is option-82-capable, it can use the remote ID, the circuit ID, or both to assign IP addresses and implement policies, such as restricting the number of IP addresses that can be assigned to a single remote ID or circuit ID. Then the DHCP server echoes the option-82 field in the DHCP reply.
- The DHCP server unicasts the reply to the switch if the request was relayed to the server by the switch. The switch verifies that it originally inserted the option-82 data by inspecting the remote ID and possibly the circuit ID fields. The switch removes the option-82 field and forwards the packet to the switch port that connects to the DHCP client that sent the DHCP request.

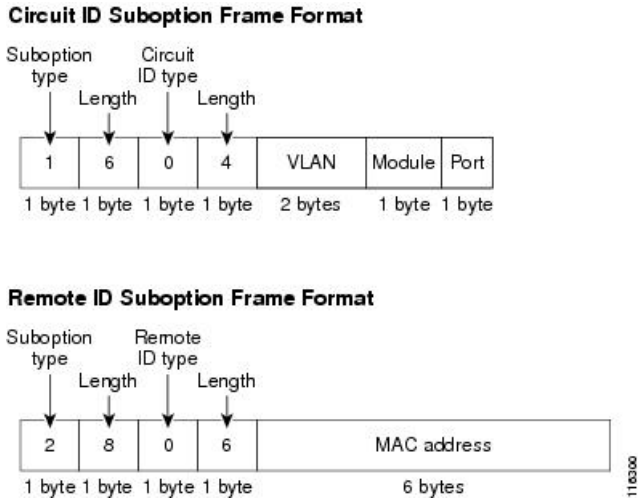
In the default suboption configuration, when the described sequence of events occurs, the values in these fields do not change (see the illustration,*Suboption Packet Formats*):

- Circuit-ID suboption fields
  - Suboption type
  - Length of the suboption type
  - Circuit-ID type
  - Length of the circuit-ID type
- Remote-ID suboption fields
  - Suboption type
  - Length of the suboption type
  - Remote-ID type
  - Length of the remote-ID type

In the port field of the circuit ID suboption, the port numbers start at 3. For example, on a switch with 24 10/100/1000 ports and four small form-factor pluggable (SFP) module slots, port 3 is the Gigabit Ethernet 1/0/1 port, port 4 is the Gigabit Ethernet 1/0/2 port, and so forth. Port 27 is the SFP module slot Gigabit Ethernet1/0/25, and so forth.

The illustration, *Suboption Packet Formats*. shows the packet formats for the remote-ID suboption and the circuit-ID suboption when the default suboption configuration is used. For the circuit-ID suboption, the module number corresponds to the switch number in the stack. The switch uses the packet formats when you globally enable DHCP snooping and enter the ip dhcp snooping information option global configuration command.

Figure 4: Suboption Packet Formats

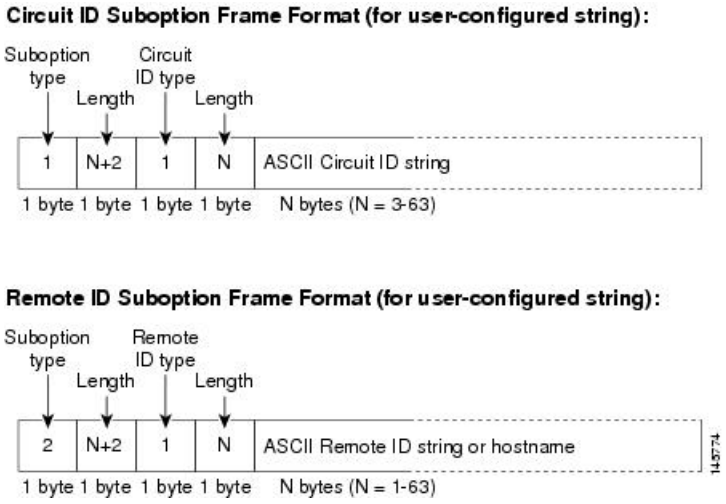


The illustration, *User-Configured Suboption Packet Formats*, shows the packet formats for user-configured remote-ID and circuit-ID suboptions The switch uses these packet formats when DHCP snooping is globally enabled and when the **ip dhcp snooping information option format remote-id** global configuration command and the**ip dhcp snooping vlan information option format-type circuit-id string** interface configuration command are entered.

The values for these fields in the packets change from the default values when you configure the remote-ID and circuit-ID suboptions:

- Circuit-ID suboption fields
  - The circuit-ID type is 1.
  - The length values are variable, depending on the length of the string that you configure.
- Remote-ID suboption fields
  - The remote-ID type is 1.
  - The length values are variable, depending on the length of the string that you configure.

Figure 5: User-Configured Suboption Packet Formats



Cisco IOS DHCP Server Database

During the DHCP-based autoconfiguration process, the designated DHCP server uses the Cisco IOS DHCP server database. It has IP addresses, address bindings, and configuration parameters, such as the boot file.

An address binding is a mapping between an IP address and a MAC address of a host in the Cisco IOS DHCP server database. You can manually assign the client IP address, or the DHCP server can allocate an IP address from a DHCP address pool.

DHCP Snooping Binding Database

When DHCP snooping is enabled, the switch uses the DHCP snooping binding database to store information about untrusted interfaces. The database can have up to 64,000 bindings.

Each database entry (binding) has an IP address, an associated MAC address, the lease time (in hexadecimal format), the interface to which the binding applies, and the VLAN to which the interface belongs. The database agent stores the bindings in a file at a configured location. At the end of each entry is a checksum that accounts for all the bytes from the start of the file through all the bytes associated with the entry. Each entry is 77 bytes, followed by a space, the checksum value, and the EOL symbol.

To keep the bindings when the switch reloads, you must use the DHCP snooping database agent. If the agent is disabled, dynamic ARP inspection or IP source guard is enabled, and the DHCP snooping binding database has dynamic bindings, the switch loses its connectivity. If the agent is disabled and only DHCP snooping is enabled, the switch does not lose its connectivity, but DHCP snooping might not prevent DHCP spoofing attacks.

When reloading, the switch reads the binding file to build the DHCP snooping binding database. The switch updates the file when the database changes.

When a switch learns of new bindings or when it loses bindings, the switch immediately updates the entries in the database. The switch also updates the entries in the binding file. The frequency at which the file is updated is based on a configurable delay, and the updates are batched. If the file is not updated in a specified time (set by the write-delay and abort-timeout values), the update stops.

This is the format of the file with bindings:

```
<initial-checksum>
TYPE DHCP-SNOOPING
VERSION 1
BEGIN
<entry-1> <checksum-1>
<entry-2> <checksum-1-2>
...
...
<entry-n> <checksum-1-2-..-n>
END
```

Each entry in the file is tagged with a checksum value that the switch uses to verify the entries when it reads the file. The initial-checksum entry on the first line distinguishes entries associated with the latest file update from entries associated with a previous file update.

This is an example of a binding file:

```
3ebe1518
TYPE DHCP-SNOOPING
VERSION 1
BEGIN
10.1.1.1 512 001.0001.0005 3EBE2881 Gi1/1                                   e5e1e733
10.1.1.1 512 001.0001.0002 3EBE2881 Gi1/1                                   4b3486ec
10.1.1.1 1536 001.0001.0004 3EBE2881 Gi1/1                                  f0e02872
10.1.1.1 1024 001.0001.0003 3EBE2881 Gi1/1                                  ac41adf9
10.1.1.1 1 001.0001.0001 3EBE2881 Gi1/1                                     34b3273e
END
```

When the switch starts and the calculated checksum value equals the stored checksum value, the switch reads entries from the binding file and adds the bindings to its DHCP snooping binding database. The switch ignores an entry when one of these situations occurs:

- The switch reads the entry and the calculated checksum value does not equal the stored checksum value. The entry and the ones following it are ignored.
- An entry has an expired lease time (the switch might not remove a binding entry when the lease time expires).
- The interface in the entry no longer exists on the system.
- The interface is a routed interface or a DHCP snooping-trusted interface.

Default DHCP Snooping Configuration

Table 1: Default DHCP Configuration

Feature	Default Setting
DHCP server	Enabled in Cisco IOS software, requires configuration[1]
DHCP relay agent	Enabled[2]
DHCP packet forwarding address	None configured
Checking the relay agent information	Enabled (invalid messages are dropped)

Feature	Default Setting
DHCP relay agent forwarding policy	Replace the existing relay agent information
DHCP snooping enabled globally	Disabled
DHCP snooping information option	Enabled
DHCP snooping option to accept packets on untrusted input interfaces[3]	Disabled
DHCP snooping limit rate	None configured
DHCP snooping trust	Untrusted
DHCP snooping VLAN	Disabled
DHCP snooping MAC address verification	Enabled
Cisco IOS DHCP server binding database	Enabled in Cisco IOS software, requires configuration. **Note** The switch gets network addresses and configuration parameters only from a device configured as a DHCP server.
DHCP snooping binding database agent	Enabled in Cisco IOS software, requires configuration. This feature is operational only when a destination is configured.

1 The switch responds to DHCP requests only if it is configured as a DHCP server.
2 The switch relays DHCP packets only if the IP address of the DHCP server is configured on the SVI of the DHCP client.
3 Use this feature when the switch is an aggregation switch that receives packets with option-82 information from an edge switch.

DHCP Snooping Configuration Guidelines

- If a switch port is connected to a DHCP server, configure a port as trusted by entering the **ip dhcp snooping trust interface** configuration command.
- If a switch port is connected to a DHCP client, configure a port as untrusted by entering the **no ip dhcp snooping trust** interface configuration command.
- You can display DHCP snooping statistics by entering the **show ip dhcp snooping statistics** user EXEC command, and you can clear the snooping statistics counters by entering the **clear ip dhcp snooping statistics** privileged EXEC command.

DHCP Server Port-Based Address Allocation

DHCP server port-based address allocation is a feature that enables DHCP to maintain the same IP address on an Ethernet switch port regardless of the attached device client identifier or client hardware address.

When Ethernet switches are deployed in the network, they offer connectivity to the directly connected devices. In some environments, such as on a factory floor, if a device fails, the replacement device must be working immediately in the existing network. With the current DHCP implementation, there is no guarantee that DHCP would offer the same IP address to the replacement device. Control, monitoring, and other software expect a stable IP address associated with each device. If a device is replaced, the address assignment should remain stable even though the DHCP client has changed.

When configured, the DHCP server port-based address allocation feature ensures that the same IP address is always offered to the same connected port even as the client identifier or client hardware address changes in the DHCP messages received on that port. The DHCP protocol recognizes DHCP clients by the client identifier option in the DHCP packet. Clients that do not include the client identifier option are identified by the client hardware address. When you configure this feature, the port name of the interface overrides the client identifier or hardware address and the actual point of connection, the switch port, becomes the client identifier.

In all cases, by connecting the Ethernet cable to the same port, the same IP address is allocated through DHCP to the attached device.

The DHCP server port-based address allocation feature is only supported on a Cisco IOS DHCP server and not a third-party server.

Default Port-Based Address Allocation Configuration

By default, DHCP server port-based address allocation is disabled.

Port-Based Address Allocation Configuration Guidelines

- By default, DHCP server port-based address allocation is disabled.
- To restrict assignments from the DHCP pool to preconfigured reservations (unreserved addresses are not offered to the client and other clients are not served by the pool), you can enter the **reserved-only** DHCP pool configuration command.

How to Configure DHCP

Configuring the DHCP Server

The switch can act as a DHCP server. If DHCP server for DHCP clients with management ports are used, both DHCP pool and the corresponding interface must be configured using the Management VRF.

Configuring the DHCP Relay Agent

Follow these steps to enable the DHCP relay agent on the switch:

Procedure

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service dhcp** **Example:** `Device(config)# service dhcp`	Enables the DHCP server and relay agent on your switch. By default, this feature is enabled.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

What to do next

- Checking (validating) the relay agent information
- Configuring the relay agent forwarding policy

Specifying the Packet Forwarding Address

If the DHCP server and the DHCP clients are on different networks or subnets, you must configure the switch with the **ip helper-address** *address* interface configuration command. The general rule is to configure the command on the Layer 3 interface closest to the client. The address used in the **ip helper-address** command can be a specific DHCP server IP address, or it can be the network address if other DHCP servers are on the destination network segment. Using the network address enables any DHCP server to respond to requests.

Perform these steps to specify the packet forwarding address:

Procedure

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

	Command or Action	Purpose
Step 3	**interface vlan** *vlan-id* **Example:** `Device(config)# interface vlan 1`	Creates a switch virtual interface by entering a VLAN ID, and enters interface configuration mode.
Step 4	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 192.108.1.27 255.255.255.0`	Configures the interface with an IP address and an IP subnet.
Step 5	**ip helper-address** *address* **Example:** `Device(config-if)# ip helper-address 172.16.1.2`	Specifies the DHCP packet forwarding address. • The helper address can be a specific DHCP server address, or it can be the network address if other DHCP servers are on the destination network segment. Using the network address enables other servers to respond to DHCP requests. • If you have multiple servers, you can configure one helper address for each server.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 7	Use one of the following: • **interface range** *port-range* • **interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/2`	Configures multiple physical ports that are connected to the DHCP clients, and enters interface range configuration mode. or Configures a single physical port that is connected to the DHCP client, and enter interface configuration mode.
Step 8	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Defines the VLAN membership mode for the port.
Step 9	**switchport access vlan** *vlan-id* **Example:**	Assigns the ports to the same VLAN as configured in Step 2.

	Command or Action	Purpose
	`Device(config-if)# switchport access vlan 1`	
Step 10	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring DHCP for IPv6 Address Assignment

Default DHCPv6 Address Assignment Configuration

By default, no DHCPv6 features are configured on the switch.

DHCPv6 Address Assignment Configuration Guidelines

The following prerequisites apply when configuring DHCPv6 address assignment:

- In the following procedures, the specified interface must be one of these Layer 3 interfaces:
  - If the IPv6 address is not explicitly configured, enable IPv6 routing by using the **ipv6 enable** command.
  - DHCPv6 routing must be enabled on a Layer 3 interface.
  - SVI: A VLAN interface created by using the **interface vlan** *vlan_id* command.
  - EtherChannel port channel in Layer 3 mode: a port-channel logical interface created by using the **interface port-channel port-channel-number** command.
- The device can act as a DHCPv6 client, server, or relay agent. The DHCPv6 client, server, and relay function are mutually exclusive on an interface.

Enabling DHCPv6 Server Function (CLI)

Use the **no** form of the DHCP pool configuration mode commands to change the DHCPv6 pool characteristics. To disable the DHCPv6 server function on an interface, use the **no ipv6 dhcp server** interface configuration command.

To enable the DHCPv6 server function on an interface, perform this procedure:

Procedure

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Device(config)# ipv6 dhcp pool 7`	Enters DHCP pool configuration mode, and define the name for the IPv6 DHCP pool. The pool name can be a symbolic string (such as Engineering) or an integer (such as 0).
Step 4	**address prefix** *IPv6-prefix* {**lifetime**} {*t1 t1* | **infinite**} **Example:** `Device(config-dhcpv6)# address prefix 2001:1000::0/64 lifetime 3600`	(Optional) Specifies an address prefix for address assignment. This address must be in hexadecimal, using 16-bit values between colons. **lifetime** *t1 t1*—Specifies a time interval (in seconds) that an IPv6 address prefix remains in the valid state. The range is 5 to 4294967295 seconds. Specify **infinite** for no time interval.
Step 5	**link-address** *IPv6-prefix* **Example:** `Device(config-dhcpv6)# link-address 2001:1002::0/64`	(Optional) Specifies a link-address IPv6 prefix. When an address on the incoming interface or a link-address in the packet matches the specified IPv6 prefix, the server uses the configuration information pool. This address must be in hexadecimal, using 16-bit values between colons.
Step 6	**vendor-specific** *vendor-id* **Example:** `Device(config-dhcpv6)# vendor-specific 9`	(Optional) Enters vendor-specific configuration mode and specifies a vendor-specific identification number. This number is the vendor IANA Private Enterprise Number. The range is 1 to 4294967295.
Step 7	**suboption** *number* {**address** *IPv6-address* | **ascii** *ASCII-string* | **hex** *hex-string*} **Example:** `Device(config-dhcpv6-vs)# suboption 1 address 1000:235D::`	(Optional) Enters a vendor-specific suboption number. The range is 1 to 65535. Enter an IPv6 address, ASCII text, or a hex string as defined by the suboption parameters.
Step 8	**exit** **Example:**	Returns to DHCP pool configuration mode.

	Command or Action	Purpose
	`Device(config-dhcpv6-vs)# exit`	
Step 9	**exit** **Example:** `Device(config-dhcpv6)# exit`	Returns to global configuration mode.
Step 10	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Enters interface configuration mode, and specifies the interface to configure.
Step 11	**ipv6 dhcp server** [*poolname* | **automatic**] [**rapid-commit**] [**preference** *value*] [**allow-hint**] **Example:** `Device(config-if)# ipv6 dhcp server automatic`	Enables DHCPv6 server function on an interface. • *poolname*—(Optional) User-defined name for the IPv6 DHCP pool. The pool name can be a symbolic string (such as Engineering) or an integer (such as 0). • **automatic**—(Optional) Enables the system to automatically determine which pool to use when allocating addresses for a client. • **rapid-commit**—(Optional) Allows two-message exchange method. • **preference** *value*—(Optional) Configures the preference value carried in the preference option in the advertise message sent by the server. The range is from 0 to 255. The preference value default is 0. • **allow-hint**—(Optional) Specifies whether the server should consider client suggestions in the SOLICIT message. By default, the server ignores client hints.
Step 12	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 13	Do one of the following:	• Verifies DHCPv6 pool configuration.

	Command or Action	Purpose

	• **show ipv6 dhcp pool**
	• **show ipv6 dhcp interface**
	Example:
	`Device# show ipv6 dhcp pool`
	or
	`Device# show ipv6 dhcp interface`
		• Verifies that the DHCPv6 server function is enabled on an interface.

Step 14	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Enabling DHCPv6 Client Function

To enable the DHCPv6 client on an interface, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Enters interface configuration mode, and specifies the interface to configure.
Step 4	**ipv6 address dhcp** [**rapid-commit**] **Example:** `Device(config-if)# ipv6 address dhcp rapid-commit`	Enables the interface to acquire an IPv6 address from the DHCPv6 server. **rapid-commit**—(Optional) Allow two-message exchange method for address assignment.

	Command or Action	Purpose
Step 5	**ipv6 dhcp client request** [**vendor-specific**] **Example:** `Device(config-if)# ipv6 dhcp client request vendor-specific`	(Optional) Enables the interface to request the vendor-specific option.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show ipv6 dhcp interface** **Example:** `Device# show ipv6 dhcp interface`	Verifies that the DHCPv6 client is enabled on an interface.

Enabling the Cisco IOS DHCP Server Database

For procedures to enable and configure the Cisco IOS DHCP server database, see the "DHCP Configuration Task List" section in the "Configuring DHCP" chapter of the Cisco IOS IP Configuration Guide.

Enabling the DHCP Snooping Binding Database Agent

Beginning in privileged EXEC mode, follow these steps to enable and configure the DHCP snooping binding database agent on the switch:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip dhcp snooping database** {**flash** [*number*] **:** */filename* | **ftp://***user* **:** *password* **@** *host* */filename* | **http://**[[*username* **:** *password*] **@**] {*hostname* | *host-ip*} [*/directory*]	Specifies the URL for the database agent or the binding file by using one of these forms: • **flash**[*number*]**:**/*filename*

	Command or Action	Purpose
	/image-name.**tar** | **rcp:**//*user* @ *host*/*filename* | **scp:**//*user*@*host*/*filename* | **tftp:**//*hostfilename*} **Example:** `Device(config)# ip dhcp snooping database tftp://10.90.90.90/snooping-rp2`	• **ftp:**//*user*:*password*@*host*/*filename* • **http:**//[[*username*:*password*]@]{*hostname* | *host-ip*}[/*directory*] /*image-name*.**tar** • **rcp:**//*user*@*host*/*filename* • **scp:**//*user*@*host*/*filename* **Note** Before you configure SCP, you need to set the line console 0 transport output to *ssh* or *all*. • **tftp:**//*host*/*filename*
Step 4	**ip dhcp snooping database timeout** *seconds* **Example:** `Device(config)# ip dhcp snooping database timeout 300`	Specifies (in seconds) how long to wait for the database transfer process to finish before stopping the process. The default is 300 seconds. The range is 0 to 86400. Use 0 to define an infinite duration, which means to continue trying the transfer indefinitely.
Step 5	**ip dhcp snooping database write-delay** *seconds* **Example:** `Device(config)# ip dhcp snooping database write-delay 15`	Specifies the duration for which the transfer should be delayed after the binding database changes. The range is from 15 to 86400 seconds. The default is 300 seconds (5 minutes).
Step 6	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 7	**ip dhcp snooping binding** *mac-address* **vlan** *vlan-id ip-address* **interface** *interface-id* **expiry** *seconds* **Example:** `Device# ip dhcp snooping binding 0001.1234.1234 vlan 1 172.20.50.5 interface gigabitethernet 1/1/0 expiry 1000`	(Optional) Adds binding entries to the DHCP snooping binding database. The *vlan-id* range is from 1 to 4904. The *seconds* range is from 1 to 4294967295. Enter this command for each entry that you add. Use this command when you are testing or debugging the switch.
Step 8	**show ip dhcp snooping database** [**detail**] **Example:**	Displays the status and statistics of the DHCP snooping binding database agent.

	Command or Action	Purpose
	`Device# show ip dhcp snooping database detail`	

Monitoring DHCP Snooping Information

Table 2: Commands for Displaying DHCP Information

show ip dhcp snooping	Displays the DHCP snooping configuration for a switch
show ip dhcp snooping binding	Displays only the dynamically configured bindings in the DHCP snooping bin also referred to as a binding table.
show ip dhcp snooping database	Displays the DHCP snooping binding database status and statistics.
show ip dhcp snooping statistics	Displays the DHCP snooping statistics in summary or detail form.
show ip source binding	Display the dynamically and statically configured bindings.

Note If DHCP snooping is enabled and an interface changes to the down state, the switch does not delete the statically configured bindings.

Enabling DHCP Server Port-Based Address Allocation

Follow these steps to globally enable port-based address allocation and to automatically generate a subscriber identifier on an interface.

Procedure

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

	Command or Action	Purpose
Step 3	**ip dhcp use subscriber-id client-id** **Example:** `Device(config)# ip dhcp use subscriber-id client-id`	Configures the DHCP server to globally use the subscriber identifier as the client identifier on all incoming DHCP messages.
Step 4	**ip dhcp subscriber-id interface-name** **Example:** `Device(config)# ip dhcp subscriber-id interface-name`	Automatically generates a subscriber identifier based on the short name of the interface. A subscriber identifier configured on a specific interface takes precedence over this command.
Step 5	**interface** *interface-type interface-number* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Specifies the interface to be configured, and enters interface configuration mode.
Step 6	**ip dhcp server use subscriber-id client-id** **Example:** `Device(config-if)# ip dhcp server use subscriber-id client-id`	Configures the DHCP server to use the subscriber identifier as the client identifier on all incoming DHCP messages on the interface.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

What to do next

After enabling DHCP port-based address allocation on the switch, use the **ip dhcp pool** global configuration command to preassign IP addresses and to associate them to clients.

Monitoring DHCP Server Port-Based Address Allocation

Table 3: Commands for Displaying DHCP Port-Based Address Allocation Information

Command	Purpose
show interface *interface id*	Displays the status and configuration of a specific interface.
show ip dhcp pool	Displays the DHCP address pools.
show ip dhcp binding	Displays address bindings on the Cisco IOS DHCP server.

Feature History for DHCP

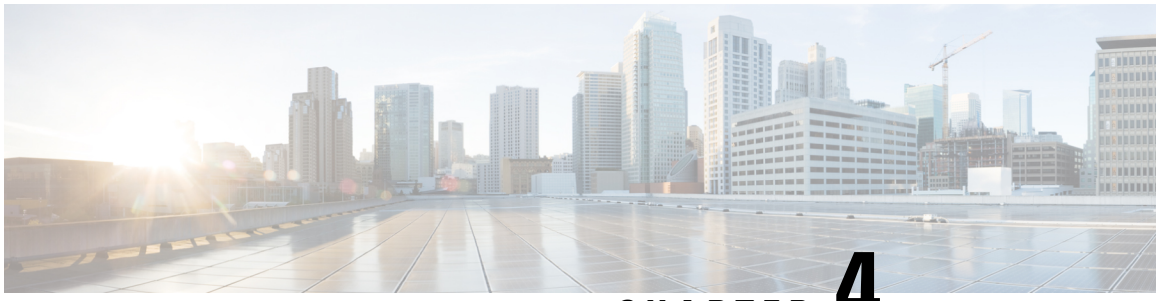
This table provides release and related information for the features explained in this module.

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

Table 4: New Feature History

Release	Feature	Feature Information
Cisco IOS XE Gibraltar 16.11.1	DHCP	DHCP provides configuration parameters to Internet hosts. DHCP consists of two components: a protocol for delivering host-specific configuration parameters from a DHCP Server to a host and a mechanism for allocating network addresses to hosts. DHCP is built on a client/server model, where designated DHCP Server hosts allocate network addresses and deliver configuration parameters to dynamically configured hosts.
Cisco IOS XE Gibraltar 16.11.1	DHCP Client Option 12	The DHCP Client Option 12 feature specifies the hostname of the client. While acquiring an IP address for an interface from the Dynamic Host Configuration Protocol (DHCP) server, if the client device receives the DHCP Hostname option inside the response, the hostname from that option is set. DHCP is used by DHCP clients to obtain configuration information for operation in an IP network.
Cisco IOS XE Cupertino 17.7.1	DHCP	This feature was implemented on Cisco Catalyst 9600 Series Supervisor 2 Module (C9600X-SUP-2).
Cisco IOS XE Cupertino 17.9.1	DHCP Snooping and Local SPAN	DHCP Snooping and Local SPAN can be configured on the same VLAN for non-SDA deployments.

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/



CHAPTER 4

DHCP Gleaning

This section provides information about DHCP Gleaning.

- Prerequisites for DHCP Gleaning, on page 47
- Information About DHCP Gleaning, on page 47
- Configuring an Interface as a Trusted or an Untrusted Source for DHCP Gleaning, on page 48
- Example: Configuring an Interface as a Trusted or an Untrusted Source for DHCP Gleaning, on page 49
- Additional References for DHCP Gleaning, on page 50
- Feature History for DHCP Gleaning, on page 50

Prerequisites for DHCP Gleaning

- Ensure that the interface to be configured is a Layer 2 interface.
- Ensure that global snooping is enabled.

Information About DHCP Gleaning

The following sections provide information about DHCP gleaning.

Overview of DHCP Gleaning

Gleaning helps extract location information from Dynamic Host Configuration Protocol (DHCP) messages when messages are forwarded by a DHCP relay agent; the process is a completely passive snooping functionality that neither blocks nor modifies DHCP packets. Additionally, gleaning helps to differentiate an untrusted device port that is connected to an end user from a trusted port connected to a DHCP server.

DHCP gleaning is a read–only DHCP snooping functionality that allows components to register and glean only DHCP version 4 packets. When you enable DHCP gleaning, it does a read-only snooping on all active interfaces on which DHCP snooping is disabled. You can add a secondary VLAN to a private VLAN. When add a secondary VLAN to a private VLAN, ensure that gleaning is enabled on the secondary VLAN, even though snooping is disabled on the primary VLAN. By default, the gleaning functionality is disabled. However, when you enable a device sensor, DHCP gleaning is automatically enabled.

DHCP Snooping

Dynamic Host Configuring Protocol (DHCP) snooping is a security feature that acts like a firewall between untrusted hosts and trusted DHCP servers. The DHCP snooping feature performs the following activities:

- Validates DHCP messages received from untrusted sources and filters out invalid messages.
- Rate-limits DHCP traffic from trusted and untrusted sources.
- Builds and maintains the DHCP snooping binding database, which contains information about untrusted hosts with leased IP addresses.
- Utilizes the DHCP snooping binding database to validate subsequent requests from untrusted hosts.

Other security features, such as dynamic Address Resolution Protocol (ARP) inspection (DAI), also uses information stored in the DHCP snooping binding database.

DHCP snooping is enabled on a per-VLAN basis. By default, the feature is inactive on all VLANs. You can enable the feature on a single VLAN or on a range of VLANs.

Configuring an Interface as a Trusted or an Untrusted Source for DHCP Gleaning

You can enable or disable DHCP gleaning on a device. You can configure an interface as a trusted or untrusted source of DHCP messages. Verify that no DHCP packets are dropped when DHCP gleaning is enabled on an untrusted interface or on a device port.

Note By default, DHCP gleaning is disabled.

You can configure DHCP trust on the following types of interfaces:

- Layer 2 Ethernet interfaces
- Layer 2 port-channel interfaces

Note By default, all interfaces are untrusted.

Procedure

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip dhcp snooping glean** **Example:** `Device(config)# ip dhcp snooping glean`	Enables DHCP gleaning on an interface.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitEthernet 1/0/1`	Enters interface configuration mode, where *type number* is the Layer 2 Ethernet interface which you want to configure as trusted or untrusted for DHCP snooping.
Step 5	[**no**] **ip dhcp snooping trust** **Example:** `Device(config-if)# ip dhcp snooping trust`	Configures the interface as a trusted interface for DHCP snooping. The **no** option configures the port as an untrusted interface.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 7	**show ip dhcp snooping statistics** **Example:** `Device# show ip dhcp snooping statistics`	Displays packets that were dropped on the device port configured as an untrusted interface.
Step 8	**show ip dhcp snooping** **Example:** `Device# show ip dhcp snooping`	Displays DHCP snooping configuration information, including information about DHCP gleaning.

Example: Configuring an Interface as a Trusted or an Untrusted Source for DHCP Gleaning

This example shows how to enable Dynamic Host Configuration Protocol (DHCP) gleaning and configure an interface as a trusted interface:

```
Device> enable
Device# configure terminal
Device(config)# ip dhcp snooping glean
Device(config)# interface gigabitEthernet 1/0/1
Device(config-if)# ip dhcp snooping trust
Device(config-if)# end
```

Additional References for DHCP Gleaning

Standards and RFCs

Standard/RFC	Title
RFC-2131	*Dynamic Host Configuration Protocol*
RFC-4388	*DHCP Leasequery*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

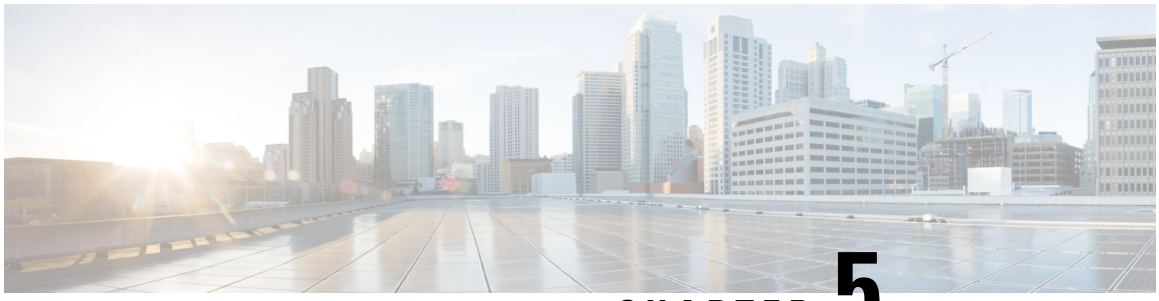
Feature History for DHCP Gleaning

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 Dublin 17.10.1	DHCP Gleaning	DHCP gleaning is a read–only DHCP snooping functionality that allows components to register and glean only DHCP version 4 packets.

Use the Cisco Feature Navigator to find information about platform and software image support.



CHAPTER 5

DHCP Options Support

- Restrictions for DHCP Options Support, on page 51
- Information About DHCP Options Support, on page 51
- Configuring DHCP Snooping on Private VLANs, on page 52
- Example: Mapping Private-VLAN Associations , on page 54
- Configuration Examples for DHCP Options Support, on page 55
- Feature History for DHCP Options Support, on page 55

Restrictions for DHCP Options Support

When DHCP snooping is configured on a primary VLAN, you cannot configure snooping with different settings on any of its secondary VLANs. You must configure DHCP snooping for all associated VLANs on the primary VLAN. If DHCP snooping is not configured on the primary VLAN and you try to configure it on the secondary VLAN, for example, VLAN 200, this message appears:

```
2w5d:%DHCP_SNOOPING-4-DHCP_SNOOPING_PVLAN_WARNING:DHCP Snooping configuration may not take
 effect
on secondary vlan 200. DHCP Snooping configuration on secondary vlan is derived from its
primary vlan.
```

You can use the **show ip dhcp snooping** command to display all VLANs, both primary and secondary, that have DHCP snooping enabled.

Information About DHCP Options Support

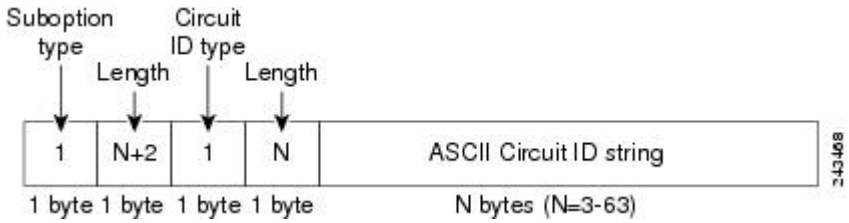
DHCP Option 82 Configurable Circuit ID and Remote ID Overview

The DHCP Option 82 Configurable Circuit ID and Remote ID feature enhances validation security by allowing you to determine what information is provided in the Option 82 Remote ID and Option 82 Circuit ID suboptions.

You can enable DHCP snooping on private VLANs. When DHCP snooping is enabled, the configuration is propagated to both a primary VLAN and its associated secondary VLANs. When DHCP snooping is enabled on a primary VLAN, it is also enabled on its secondary VLANs.

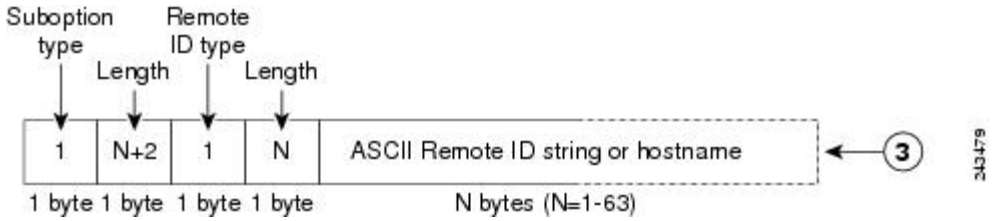
The figure below shows the packet format used when DHCP snooping is globally enabled and the **ip dhcp snooping information option** global configuration command is entered with the Circuit ID suboption.

Figure 6: Suboption Packet Formats, Circuit ID Specified



The figure below shows the packet format used when DHCP snooping is globally enabled and the **ip dhcp snooping information option** global configuration command is entered with the Remote ID suboption.

Figure 7: Suboption Packet Formats, Remote ID Specified



DHCP Client Option 12

The DHCP Client Option12 feature specifies the hostname of the client. While acquiring an IP address for an interface from the Dynamic Host Configuration Protocol (DHCP) server, if the client device receives the DHCP Hostname option inside the response, the hostname from that option is set. DHCP is used by DHCP clients to obtain configuration information for operation in an IP network.

Configuration parameters and other control information are carried in tagged data items that are stored in the options field of a DHCP message. The DHCP client provides flexibility by allowing Option 12 to be configured for a DHCP client.

Option 12 specifies the name of the client. The name might or might not be qualified with the local domain.

Configuring DHCP Snooping on Private VLANs

Perform these tasks to configure DHCP snooping on private primary and secondary VLANs:

- Configure a private, primary VLAN.
- Associate with it an isolated VLAN.
- Create an SVI interface for the primary VLAN, and associate it with the appropriate loopback IP and helper address.
- Enable DHCP snooping on the primary VLAN, which also enables it on the associated VLAN.

Note You must also configure a server to assign the IP address, a DHCP pool, and a relay route so that snooping can be effective.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan** *vlan-id* **Example:** `Device(config)# vlan 70`	Enters VLAN configuration mode for the named private VLAN.
Step 4	**private-vlan primary** **Example:** `Device(config-vlan)# private-vlan primary`	Designates the VLAN as the primary private VLAN.
Step 5	**private-vlan association** *secondary-vlan-list* **Example:** `Device(config-vlan)# private-vlan association 7`	Configures private VLANs (PVLANs) and the association between a PVLAN and a secondary VLAN.
Step 6	**exit** **Example:** `Device(ocnfig-vlan)# exit`	Exits VLAN configuration mode and returns to global configuration mode.
Step 7	**vlan** *vlan_ID* **Example:** `Device(config)# vlan 7`	Enters VLAN configuration mode for the named private VLAN. • In this example, the associated secondary VLAN is vlan 7.
Step 8	**private-vlan isolated** **Example:** `Device(config-vlan)# private-vlan isolated`	Designates the VLAN as an isolated private VLAN.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config-vlan)# exit`	Exits VLAN configuration mode and returns to global configuration mode.
Step 10	**interface vlan** *primary-vlan_id* **Example:** `Device(config)# interface vlan 70`	Creates a dynamic Switch Virtual Interface (SVI) on the primary VLAN, and enters interface configuration mode.
Step 11	**ip unnumbered loopback** **Example:** `Device(config-if)# ip unnumbered loopback1`	Specifies IP unnumbered loopback.
Step 12	**private-vlan mapping** [*secondary-vlan-list* | **add** *secondary-vlan-list*| **remove** *secondary-vlan-list*] **Example:** `Device(config-if)# private-vlan mapping 7`	Creates a mapping between the primary and the secondary VLANs so that they share the same primary VLAN SVI.
Step 13	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 14	**ip dhcp snooping vlan** *primary-vlan_id* **Example:** `Device(config)# ip dhcp snooping vlan 70`	Enables DHCP snooping on the primary and associated VLANs.
Step 15	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Example: Mapping Private-VLAN Associations

The following interface configuration example shows how to map the private-VLAN associations. The user-configurable circuit ID “aabb11” is inserted on the secondary VLAN, vlan 7.

```
Device> enable
Device# configure terminal
```

```
Device(config-if)# interface GigabitEthernet 9/0/1
Device(config-if)# switchport
Device(config-if)# switchport private-vlan host-association 70 7
Device(config-if)# switchport mode private-vlan host
Device(config-if)# no mls qos trust
Device(config-if)# spanning-tree portfast
Device(config-if)# exit
Device(config)# ip dhcp snooping vlan 7 information option format-type circuit-id string
aabb11
Device(config)# end
```

The following example shows how to define a DHCP class "C1" and specify the hex string of the corresponding class at the server by using the hex string that matches the circuit-ID value entered in the interface configuration example. That is, the hex string 00000000000000000000000000000006616162623131 mask ffffffffffffffffffffffffffff0000000000000 matches the circuit ID aabb11.

```
Device> enable
Device# configure terminal
Device(config)# ip dhcp class C1
Device(config-dhcp-class)# relay agent information
Device(config-dhcp-class-relayinfo)# relay-information hex
00000000000000000000000000000006616162623131
mask ffffffffffffffffffffffffffff0000000000000
Device(config-dhcp-class-relayinfo)# end
```

Configuration Examples for DHCP Options Support

Feature History for DHCP Options Support

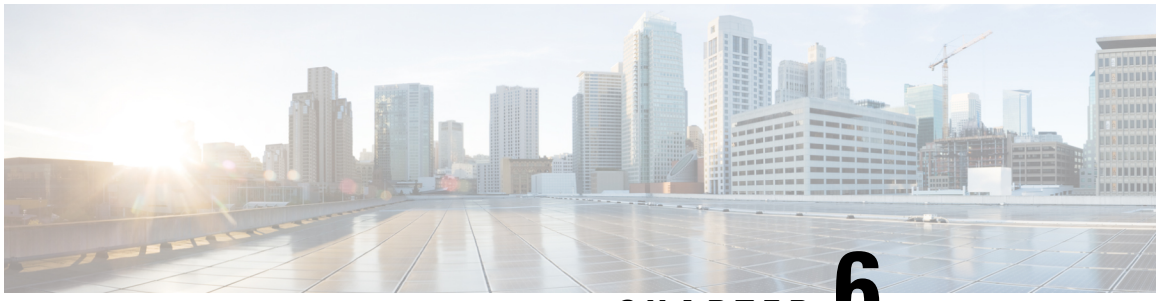
This table provides release and related information for the features explained in this module.

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 Gibraltar 16.11.1	DHCP Client Option 12	The DHCP Client Option 12 feature specifies the hostname of the client. While acquiring an IP address for an interface from the Dynamic Host Configuration Protocol (DHCP) server, if the client device receives the DHCP Hostname option inside the response, the hostname from that option is set. DHCP is used by DHCP clients to obtain configuration information for operation in an IP network.
	DHCP Option 82 Configurable Circuit ID and Remote ID	Provides naming choices in the Option 82 Remote ID and Option 82 Circuit ID suboptions.

Release	Feature	Feature Information
Cisco IOS XE Cupertino 17.7.1	DHCP Client Option 12, Option 82 Configurable Circuit ID and Remote ID	This feature was implemented on the Cisco Catalyst 9600 Series Supervisor 2 Module (C9600X-SUP-2).

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/



CHAPTER 6

DHCPv6 Options Support

- Information About DHCPv6 Options Support, on page 57
- How to Configure DHCPv6 Options Support, on page 59
- Example: Configuring CAPWAP Access Points, on page 61
- Verifying DHCPv6 Options Support, on page 62
- Additional References for DHCPv6 Options Support, on page 62
- Feature History for DHCPv6 Options Support, on page 63

Information About DHCPv6 Options Support

CAPWAP Access Controller DHCPv6 Option

The Control And Provisioning of Wireless Access Points (CAPWAP) protocol allows lightweight access points to use DHCPv6 to discover a wireless controller to which it can connect. CAPWAP is a standard, interoperable protocol that enables a controller to manage a collection of wireless access points.

Wireless access points use the DHCPv6 option 52 (RFC 5417) to supply the IPv6 management interface addresses of the primary, secondary, and tertiary wireless controllers.

Both stateless and stateful DHCPv6 addressing modes are supported. In stateless mode, access points obtain IPv6 address using the Stateless Address Auto Configuration (SLAAC), while additional network information (not obtained from router advertisements) is obtained from a DHCPv6 server. In stateful mode, access points obtain both IPv6 addressing and additional network information exclusively from the DHCPv6 server. In both modes, a DHCPv6 server is required to provide option 52 if Wireless Controller discovery using DHCPv6 is required.

When the MAX_PACKET_SIZE exceeds 15, and option 52 is configured, the DHCPv6 server does not send DHCP packets.

DNS Search List Option

DNS Search List (DNSSL) is a list of Domain Name System (DNS) suffix domain names used by IPv6 hosts when they perform DNS query searches for short, unqualified domain names. The DNSSL option contains one or more domain names. All domain names share the same lifetime value, which is the maximum time in seconds over which this DNSSL may be used. If different lifetime values are required, multiple DNSSL options can be used. There can be a maximum of 5 DNSSLs.

DHCP messages with long DNSSL names are discarded by the device.

Note If DNS information is available from multiple Router Advertisements (RAs) and/or from DHCP, the host must maintain an ordered list of this DNS information.

RFC 6106 specifies IPv6 Router Advertisement (RA) options to allow IPv6 routers to advertise a DNS Search List (DNSSL) to IPv6 hosts for an enhanced DNS configuration.

The DNS lifetime range should be between the maximum RA interval and twice the maximum RA interval, as displayed in the following example:

```
(max ra interval) <= dns lifetime <= (2*(max ra interval))
```

The maximum RA interval can have a value between 4 and 1800 seconds (the default is 240 seconds). The following example shows an out-of-range lifetime:

```
Device(config-if)# ipv6 nd ra dns-search-list sss.com 3600
! Lifetime configured out of range for the interface that has the  default maximum RA
interval.!
```

DHCPv6 Client Link-Layer Address Option

The DHCPv6 Client Link-Layer Address Option (RFC 6939) defines an optional mechanism and the related DHCPv6 option to allow first-hop DHCPv6 relay agents (relay agents that are connected to the same link as the client) to provide the client's link-layer address in DHCPv6 messages that are sent towards the server.

The Client Link-Layer Address option is only exchanged between relay agents and servers. DHCPv6 clients are not aware of the use of the Client Link-Layer Address option. The DHCPv6 client must not send the Client Link-Layer Address option, and must ignore the Client Link-Layer Address option if received.

Each DHCPv6 client and server is identified by a DHCP unique identifier (DUID). The DUID is carried in the client identifier and server identifier options. The DUID is unique across all DHCP clients and servers, and it is stable for any specific client or server. DHCPv6 uses DUIDs based on link-layer addresses for both the client and server identifier. The device uses the MAC address from the lowest-numbered interface to form the DUID. The network interface is assumed to be permanently attached to the device.

DHCP Relay Agent

A DHCP relay agent is a Layer 3 device that forwards DHCP packets between clients and servers. Relay agents forward requests and replies between clients and servers when they are not on the same physical subnet. Relay agent forwarding is different from the normal Layer 2 forwarding, in which IP datagrams are switched transparently between networks. Relay agents receive DHCP messages and generate new DHCP messages to send on output interfaces.

DHCPv6 Relay Agent

A DHCPv6 relay agent, which may reside on the client’s link, is used to relay messages between the client and the server. The DHCPv6 relay agent operation is transparent to the client. A DHCPv6 client locates a

DHCPv6 server using a reserved, link-scoped multicast address. For direct communication between the DHCPv6 client and the DHCPv6 server, both of them must be attached to the same link. However, in some situations where ease of management, economy, or scalability is a concern, it is desirable to allow a DHCPv6 client to send a message to a DHCPv6 server that is not connected to the same link.

DHCPv6 Relay Interface-Id Option

A DHCPv6 relay agent adds an Interface-Id option in the upstream DHCPv6 message. The Interface-Id option serves to identify the interface on which the client is connected. This information is used by the DHCPv6 relay agent while forwarding the downstream DHCPv6 message to the DHCPv6 client.

In a scenario where a Switch Virtual Interface (SVI) is configured to act as a relay agent, the Interface-Id option does not carry the physical interface details of the client interface. The Interface-Id option contains only the VLAN number of the client interface. The DHCPv6 server cannot identify which client sent the packet. The server cannot assign IPv6 addressess and policies to the packet.

Starting with the Cisco IOS XE Dublin 17.12.1 release, when an SVI acts as a relay agent the Interface-Id option will contain the physical interface details of the client interface. The physical interface details are included along with the VLAN number which is included by default. The new data is added as a sub-option. This makes it backward compatible as well as easily extensible.

The following is an example of the Interface-Id format before the physical interface details of the client interface are included.

```
Interface-Id String: 0x0105566C313030
Sub-op code: 01
Length :05
data: 566C313030 (Vlan100)
```

The following is an example of the Interface-Id format after the physical interface details of the client interface are included.

```
Interface-Id String: 0x0105566C31303002074769302F312F30
New sub option to include physical interface name
Sub-op code: 02
length:07
data:4769302F312F30 (Gi0/1/0)
```

How to Configure DHCPv6 Options Support

This section provides information about how to configure DHCPv6 options support:

Configuring CAPWAP Access Points

Procedure

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Device(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 server configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**capwap-ac address** *ipv6-address* **Example:** `Device(config-dhcpv6)# capwap-ac address 2001:DB8::1`	Configures CAPWAP access controller address.
Step 5	**end** **Example:** `Device(config-dhcpv6)# end`	Exits DHCPv6 pool configuration mode and returns to privileged EXEC mode.

Configuring DNS Search List Using IPv6 Router Advertisement Options

Perform this task to configure the DNS search list using IPv6 router advertisement options:

Note The domain name configuration should follow RFC 1035. If not, the configuration will be rejected. For example, the following domain name configuration will result in an error:

```
Device(config-if)# ipv6 nd ra dns-search-list domain example.example.com infinite-lifetime
```

Note The **ipv6 nd ra dns-search-list domain** command can only be configured on physical interfaces that are configured as routed ports in layer 3 mode. This is done by running the **no switchport** command in interface configuration mode.

Use the **no ipv6 nd ra dns-search-list domain** *domain-name* command in interface configuration mode to delete a single DNS search list under an interface.

Procedure

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-type interface-number* **Example:** `Device(config)# interface GigabitEthernet 0/2/0`	Configures an interface and enters interface configuration mode.
Step 4	**no switchport** **Example:** `Device(config-if)# no switchport`	For physical ports only, enters Layer 3 mode.
Step 5	**ipv6 nd prefix** *ipv6-prefix/prefix-length* **Example:** `Device(config-if)# ipv6 nd prefix 2001:DB8::1/64 1111 222`	Configures IPv6 prefixes that are included in IPv6 Neighbor Discovery (ND) router advertisements.
Step 6	**ipv6 nd ra lifetime** *seconds* **Example:** `Device(config-if)# ipv6 nd ra lifetime 9000`	Configures the device lifetime value in IPv6 router advertisements on an interface.
Step 7	**ipv6 nd ra dns-search-list domain** *domain-name* [**lifetime** [*lifetime-value* | **infinite**]] **Example:** `Device(config-if)# ipv6 nd ra dns-search-list domain example.example.com lifetime infinite`	Configures the DNS search list. You can specify the life time of the search list. **Note** For releases earlier than Cisco IOS XE Giraltar 16.12.1, this command existed as **ipv6 nd ra dns search list** *list-name* **infinite-lifetime**
Step 8	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example: Configuring CAPWAP Access Points

The following example shows how to configure a CAPWAP access point:

```
Device> enable
Device# configure terminal
Device(config)# ipv6 dhcp pool pool1
Device(config-dhcpv6)# capwap-ac address 2001:DB8::1
Device(config-dhcpv6)# end
```

```
Device#
```

Verifying DHCPv6 Options Support

Verifying Option 52 Support

The following sample output from the **show ipv6 dhcp pool** command displays the DHCPv6 configuration pool information:

```
Device# show ipv6 dhcp pool

DHCPv6 pool: svr-p1
  Static bindings:
    Binding for client 000300010002FCA5C01C
      IA PD: IA ID 00040002,
        Prefix: 2001:db8::3/72
                preferred lifetime 604800, valid lifetime 2592000
      IA PD: IA ID not specified; being used by 00040001
        Prefix: 2001:db8::1/72
                preferred lifetime 240, valid lifetime 54321
        Prefix: 2001:db8::2/72
                preferred lifetime 300, valid lifetime 54333
        Prefix: 2001:db8::3/72
                preferred lifetime 280, valid lifetime 51111
  Prefix from pool: local-p1, Valid lifetime 12345, Preferred lifetime 180
  DNS server: 1001::1
  DNS server: 1001::2
  CAPWAP-AC Controller address: 2001:DB8::1
  Domain name: example1.com
  Domain name: example2.com
  Domain name: example3.com
 Active clients: 2
```

The following example shows how to enable debugging for DHCPv6:

```
Device# debug ipv6 dhcp detail

IPv6 DHCP debugging is on (detailed)
```

Additional References for DHCPv6 Options Support

Standards and RFCs

Standards/RFC	Title
RFC 6106	IPv6 Router Advertisement Options for DNS Configuration
RFC 54171	Control And Provisioning of Wireless Access Points (CAPWAP) Access Controller DHCP Option
RFC 6939	Client Link-Layer Address Option in DHCPv6

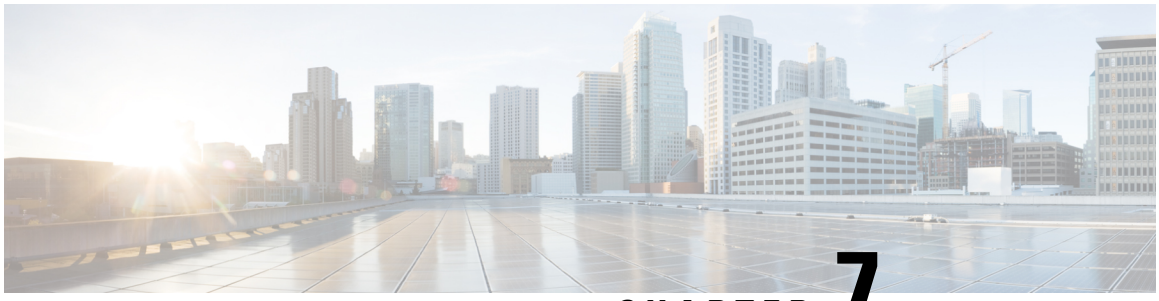
Feature History for DHCPv6 Options Support

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 Gibraltar 16.11.1	CAPWAP Access Controller DHCPv6 Option-52	The CAPWAP protocol allows lightweight access points to use DHCPv6 to discover a Wireless Controller to which it can connect. CAPWAP is a standard, interoperable protocol that enables a controller to manage a collection of wireless access points.
	DHCPv6 Client Link-Layer Address Option	The DHCPv6 Client Link-Layer Address Option (RFC 6939) defines an optional mechanism and the related DHCPv6 option to allow first-hop DHCPv6 relay agents (relay agents that are connected to the same link as the client) to provide the client's link-layer address in the DHCPv6 messages being sent towards the server.
	DNS Search List	DNS Search List (DNSSL) is a list of Domain Name System (DNS) suffix domain names used by IPv6 hosts when they perform DNS query searches for short, unqualified domain names. The DNSSL option contains one or more domain names.
Cisco IOS XE Gibraltar 16.12.1	DHCPv6 Relay Chaining and Route Insertion	DHCPv6 Relay Chaining and Route Insertion feature allows DHCPv6 messages to be relayed through multiple relay agents.
	DHCPv6 Client Link-Layer Address Option - Command Changes	The syntax of **ipv6 nd ra dns search list** command was modified to **ipv6 nd ra dns-search-list domain** . The **show ipv6 nd ra dns-search-list** command was introduced.
	IPv6 Support for RFC 6106 and RFC 5417	IPv6 support was introduced for Router Advertisement Options for DNS Configuration (RFC 6106), and Control And Provisioning of Wireless Access Points (CAPWAP) Access Controller DHCP Option (RFC 5417).
Cisco IOS XE Cupertino 17.7.1	DHCPv6 Options Support	This feature was implemented on the Cisco Catalyst 9600 Series Supervisor 2 Module (C9600X-SUP-2).

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/



CHAPTER 7

DHCPv6 Relay Source Configuration

- Restrictions for Configuring a DHCPv6 Relay Source, on page 65
- Information About DHCPv6 Relay Source Configuration, on page 65
- Configuring a DHCPv6 Relay Source, on page 66
- Example: Configuring a DHCPv6 Relay Source on an Interface, on page 67
- Additional References for DHCPv6 Relay Source Configuration, on page 68
- Feature History for DHCPv6 Relay Source Configuration, on page 68

Restrictions for Configuring a DHCPv6 Relay Source

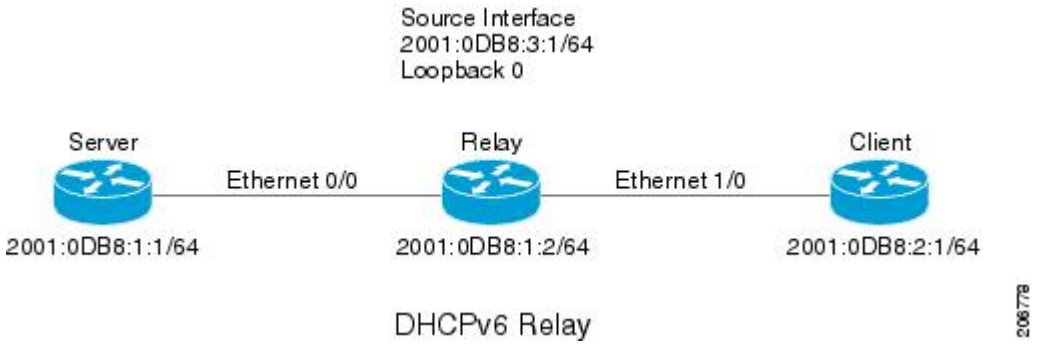
- If the configured interface is shut down, or if all of its IPv6 addresses are removed, the relay will revert to its standard behavior.
- The command line interface (CLI) will report an error if the user attempts to specify an interface that has no IPv6 addresses configured.
- The interface configuration takes precedence over the global configuration if both have been configured.

Information About DHCPv6 Relay Source Configuration

The DHCPv6 server sends its replies to the source address of relayed messages. Normally, a DHCPv6 relay uses the address of the server-facing interface used to send messages as the source. However, in some networks, it may be desirable to configure a more stable address (such as a loopback interface) and have the relay use that interface as the source address of relayed messages. The DHCPv6 Relay Source Configuration feature provides this capability.

The figure below shows a simple network with a single client, relay, and server. The relay and server communicate over 2001:DB8:1::/64, and the relay has a client-facing interface on 2001:DB8:2::/64. The relay also has a loopback interface configured with address 2001:DB8:3:1/64.

Figure 8: DHCPv6 Relay Source Configuration—Simple Network



When the relay receives a request from the client, the relay includes an address from the client-facing interface (Ethernet 1/0) in the link-address field of a relay-forward message. This address is used by the server to select an address pool. The relay then sends the relay-forward message toward the server. By default, the address of the server-facing (Ethernet 0/0) interface is used as the IPv6 source, and the server will send any reply to that address.

If the relay source interface is explicitly configured, the relay will use that interface's primary IPv6 address as the IPv6 source for messages it forwards. For example, configuring Loopback 0 as the source would cause the relay to use 2001:DB8:3:1/64 as the IPv6 source address for messages relayed toward the server.

Configuring a DHCPv6 Relay Source

Perform the following tasks to configure a DHCPv6 relay source:

Configuring a DHCPv6 Relay Source on an Interface

Perform this task to configure an interface to use as the source when relaying messages.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface loopback 0`	Specifies an interface type and number, and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ipv6 dhcp relay source-interface** *interface-type interface-number* **Example:** `Device(config-if)# ipv6 dhcp relay source-interface loopback 0`	Configures an interface to use as the source when relaying messages received on this interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring a DHCPv6 Relay Source Globally

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp relay source-interface** *interface-type interface-number* **Example:** `Device(config)# ipv6 dhcp relay source-interface loopback 0`	Configures an interface to use as the source when relaying messages.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Example: Configuring a DHCPv6 Relay Source on an Interface

The following example show how to configure the Loopback 0 interface to be used as the relay source:

```
Device> enable
Device# configure terminal
Device(config)# interface loopback 0
Device(config-if)# ipv6 dhcp relay source-interface loopback 0
Device(config-if)# end
```

Additional References for DHCPv6 Relay Source Configuration

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	*IPv6 RFCs*

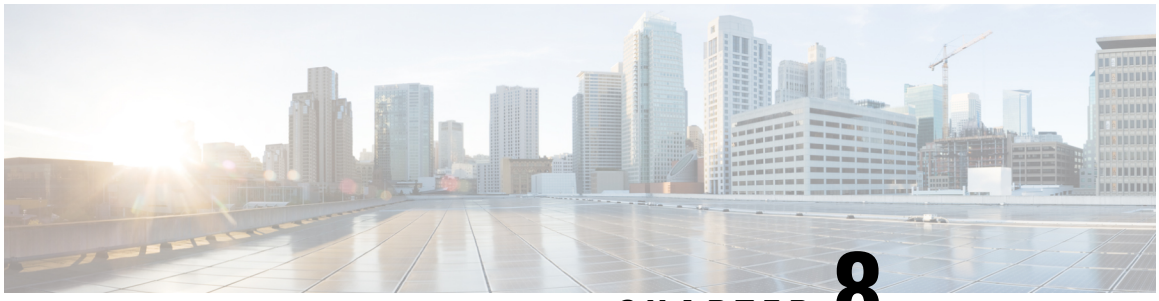
Feature History for DHCPv6 Relay Source Configuration

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 Gibraltar 16.11.1	DHCPv6 Relay Source Configuration	In some networks that use DHCPv6, it may be desirable to configure a stable address (such as a loopback interface) and have the relay use that interface as the source address of relayed messages. The DHCPv6 relay source configuration feature provides this capability.
Cisco IOS XE Cupertino 17.7.1	DHCPv6 Relay Source Configuration	This feature was implemented on the Cisco Catalyst 9600 Series Supervisor 2 Module (C9600X-SUP-2).

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/



CHAPTER 8

Configuring GRE IPv6 Tunnels

- Restrictions for GRE IPv6 Tunnels, on page 69
- Information About GRE IPv6 Tunnels, on page 69
- How to Configure GRE IPv6 Tunnels, on page 70
- Configuration Examples for GRE IPv6 Tunnels, on page 73
- Feature History for GRE IPv6 Tunnels, on page 74

Restrictions for GRE IPv6 Tunnels

- Keepalive is not supported over GRE IPv6 Tunnels, whereas it is supported over GRE IPv4 Tunnels.
- ISIS is not supported over GRE tunnels.
- Checksum is supported over GRE IPv6 Tunnels but not over GRE IPv4 Tunnels.
- MPLS over GRE IPv6 Tunnel is not supported whereas GRE IPv6 Tunnel over MPLS is supported.
- No feature interactions such as IPSec, ACL, Tunnel counters, Crypto support, Fragmentation, Cisco Discovery Protocol (CDP), QoS, GRE keepalive, etc. are supported on GRE tunnels.

Information About GRE IPv6 Tunnels

Overview of GRE IPv6 Tunnels

The GRE IPv6 Tunnels feature enables the delivery of packets from other protocols through an IPv6 network and allows the routing of IPv6 packets between private networks across public networks with globally routed IPv6 addresses.

For point-to-point GRE tunnels, each tunnel interface requires a tunnel source IPv6 address and a tunnel destination IPv6 address when being configured. All packets are encapsulated with an outer IPv6 header and a GRE header.

GRE IPv6 Tunnel Protection

GRE IPv6 tunnel protection allows devices to work as security gateways, establish IPsec tunnels between other security gateway devices, and provide crypto IPsec protection for traffic from internal networks when the traffic is sent across the public IPv6 Internet. The GRE IPv6 tunnel protection functionality is similar to the security gateway model that uses GRE IPv4 tunnel protection.

Distributed GRE Tunneling Support

Distributed GRE Tunneling allows Cisco IOS software to switch packets into and out of the Generic Routing Encapsulation (GRE) tunnels using distributed Cisco Express Forwarding (dCEF). The tunneling is performed using recursive or "double" switching techniques that are currently deployed on existing non-distributed platforms. The relevant bits are ported into this development.

Double switching is performed by the handling of the received IP packet in the existing code path until it is determined that the packet needs encapsulation or de-encapsulation. Recursively forwarding the IP packet through the IP switching path again explains the "double" aspect of the switching.

The GRE tunneling allows service providers to support a large number of tunnels by forwarding distributed tunneled packets. This feature is an extension of the non-distributed forwarding information base (FIB) forwarding paths.

Note dCEF must be explicitly enabled on the device before GRE tunneling. At the tunnel exit point, dCEF and Cisco Express Forwarding (CEF) GRE tunnels do not support reassembly of fragmented packets. Also, dCEF and CEF GRE tunnels do not support packet sequencing or check summing as defined in RFC 1721.

How to Configure GRE IPv6 Tunnels

Configuring GRE IPv6 Tunnels

Perform this task to configure a GRE tunnel on an IPv6 network. GRE tunnels can be configured to run over an IPv6 network layer and transport IPv6 and IPv4 packets through IPv6 tunnels.

Note You must enable IPv6 or configure IPv6 MTU size more than 1500 on a tunnel's exit interface to avoid receiving warning messages.

Before you begin

When GRE IPv6 tunnels are configured, IPv6 addresses are assigned to the tunnel source and the tunnel destination. The tunnel interface can have either IPv4 or IPv6 addresses. The host or device at each end of the configured tunnel must support both IPv4 and IPv6 protocol stacks.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *tunnel-number* **Example:** `Device(config)# interface tunnel 0`	Specifies a tunnel interface and number and enters interface configuration mode.
Step 4	**tunnel source** {*ipv6-address* | *interface-type* |*interface-number* } **Example:** `Device(config-if)# tunnel source ethernet 0`	Specifies the source IPv6 address or the source interface type and number for the tunnel interface. • If an interface type and number are specified, the interface must be configured with an IPv6 address. **Note** For more information on the tunnel source command, refer to the IPv6 command reference guide.
Step 5	**tunnel destination** *ipv6-address* **Example:** `Device(config-if)# tunnel destination 2001:0DB8:0C18:2::300`	Specifies the destination IPv6 address for the tunnel interface. **Note** For more information on the tunnel destination command, refer to the IPv6 command reference guide.
Step 6	**tunnel mode gre ipv6** **Example:** `Device(config-if)# tunnel mode gre ipv6`	Specifies a GRE IPv6 tunnel. **Note** The **tunnel mode gre ipv6** command specifies GRE as the encapsulation protocol for the tunnel interface. Only the syntax used in this context is displayed. For more details, see the IPv6 Command Reference.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring GRE IPv6 Tunnel Protection

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *tunnel-number* **Example:** `Device(config)# interface tunnel 0`	Specifies a tunnel interface and number and enters interface configuration mode.
Step 4	**tunnel source** {*ipv6-address* | *interface-type interface-number*} **Example:** `Device(config-if)# tunnel source ethernet 0`	Specifies the source IPv6 address or the source interface type and number for the tunnel interface. • If an interface type and number are specified, the interface must be configured with an IPv6 address. **Note** Only the syntax used in this context is displayed. For more details, see the IPv6 Command Reference.
Step 5	**tunnel destination** *ipv6-address* **Example:** `Device(config-if)# tunnel destination 2001:0DB8:0C18:2::300`	Specifies the destination IPv6 address for the tunnel interface. **Note** Only the syntax used in this context is displayed. For more details, see the IPv6 Command Reference.
Step 6	**tunnel mode gre ipv6** **Example:** `Device(config-if)# tunnel mode gre ipv6`	Specifies a GRE IPv6 tunnel. **Note** The **tunnel mode gre ipv6** command specifies GRE as the encapsulation protocol for the tunnel interface. Only the syntax used in this context is displayed. For more details, see the IPv6 Command Reference.

	Command or Action	Purpose
Step 7	**tunnel protection ipsec profile** *profile-name* **Example:** `Device(config-if)# tunnel protection ipsec profile ipsec-profile`	Associates the tunnel interface with an IPsec profile. **Note** For the *profile-name* argument, specify the IPsec profile configured in global configuration mode.
Step 8	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for GRE IPv6 Tunnels

Example: Configuring GRE IPv6 Tunnels

The following example shows how to configure a GRE tunnel over an IPv6 transport. In this example, Ethernet0/0 has an IPv6 address, and this is the source address used by the tunnel interface. The destination IPv6 address of the tunnel is specified directly. In this example, the tunnel carries both IPv4 and IS-IS traffic.

```
interface Tunnel0
 ip address 10.1.1.1 255.255.255.0
 ip router isis
 tunnel source Ethernet0/0
 tunnel destination 2001:DB8:1111:2222::1
 tunnel mode gre ipv6
!
interface Ethernet0/0
 no ip address
 ipv6 address 2001:DB8:1111:1111::1/64
!
router isis
 net 49.0001.0000.0000.000a.00
```

Example: Configuring GRE IPv6 Tunnel Protection

The following example shows how to associate the IPsec profile “ipsec-profile” with a GRE IPv6 tunnel interface. The IPsec profile is configured using the **crypto ipsec profile** command.

```
crypto ipsec profile ipsec-profile
 set transform-set ipsec-profile
!
interface Tunnel1
 ip address 192.168.1.1 255.255.255.252
 tunnel source FastEthernet2/0
 tunnel destination 10.13.7.67
 tunnel protection ipsec profile ipsec-profile
```

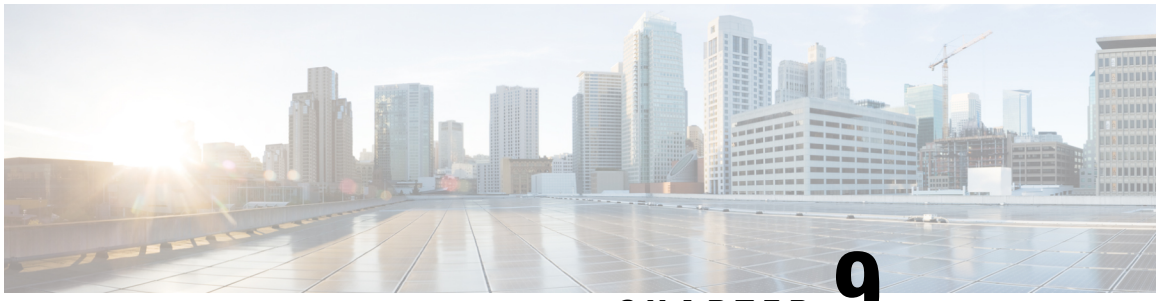
Feature History for GRE IPv6 Tunnels

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 17.1.1	GRE IPv6 Tunnels	The feature was introduced.

Use the Cisco Feature Navigator to find information about platform and software image support.



CHAPTER 9

Configuring IPv6 over IPv4 GRE Tunnels

- Restrictions for IPv6 over IPv4 GRE Tunnels, on page 75
- Information About Configuring IPv6 over IPv4 GRE Tunnels, on page 75
- Configuring GRE IPv6 Tunnels, on page 76
- Configuration Example: Tunnel Destination Address for IPv6 Tunnel, on page 78
- Additional References, on page 78
- Feature History for IPv6 over IPv4 GRE Tunnels, on page 78

Restrictions for IPv6 over IPv4 GRE Tunnels

This feature is not supported on the Cisco Catalyst 9600 Series Supervisor 2 Module.

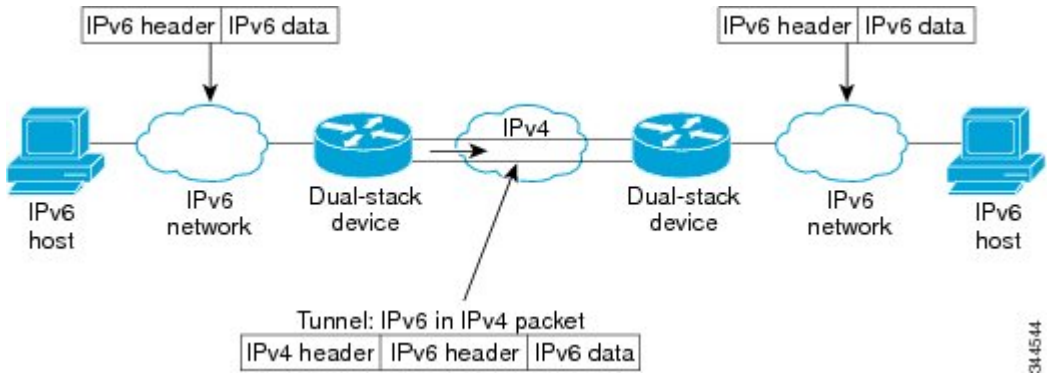
Information About Configuring IPv6 over IPv4 GRE Tunnels

The following sections provide information about configuring IPv6 over IPv4 GRE tunnels:

Overlay Tunnels for IPv6

Overlay tunneling encapsulates IPv6 packets in IPv4 packets for delivery across an IPv4 infrastructure (a core network or the figure below). By using overlay tunnels, you can communicate with isolated IPv6 networks without upgrading the IPv4 infrastructure between them. Overlay tunnels can be configured between border devices or between a border device and a host; however, both tunnel endpoints must support both the IPv4 and IPv6 protocol stacks.

Figure 9: Overlay Tunnels



Note Overlay tunnels reduce the maximum transmission unit (MTU) of an interface by 20 octets (assuming that the basic IPv4 packet header does not contain optional fields). A network that uses overlay tunnels is difficult to troubleshoot. Therefore, overlay tunnels that connect isolated IPv6 networks should not be considered a final IPv6 network architecture. The use of overlay tunnels should be considered as a transition technique toward a network that supports both the IPv4 and IPv6 protocol stacks or just the IPv6 protocol stack.

IPv6 supports GRE type of overlay tunneling. IPv6 over IPv4 GRE Tunnels can carry IPv6, Connectionless Network Service (CLNS), and many other types of packets.

GRE IPv4 Tunnel Support for IPv6 Traffic

IPv6 traffic can be carried over IPv4 GRE tunnels using the standard GRE tunneling technique that is designed to provide the services to implement any standard point-to-point encapsulation scheme. As in IPv6 manually configured tunnels, GRE tunnels are links between two points, with a separate tunnel for each link. The tunnels are not tied to a specific passenger or transport protocol but, in this case, carry IPv6 as the passenger protocol with the GRE as the carrier protocol and IPv4 or IPv6 as the transport protocol.

The primary use of GRE tunnels is for stable connections that require regular secure communication between two edge devices or between an edge device and an end system. The edge devices and the end systems must be dual-stack implementations.

Configuring GRE IPv6 Tunnels

Perform this task to configure a GRE tunnel on an IPv6 network. GRE tunnels can be configured to run over an IPv6 network layer and to transport IPv6 packets in IPv6 tunnels and IPv4 packets in IPv6 tunnels.

To configure GRE IPv6 tunnels, perform this procedure:

Before you begin

When GRE IPv6 tunnels are configured, IPv6 addresses are assigned to the tunnel source and the tunnel destination. The tunnel interface can have either IPv4 or IPv6 addresses assigned (this is not shown in the task). The host or router at each end of a configured tunnel must support both the IPv4 and IPv6 protocol stacks.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *tunnel-number* **Example:** `Device(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-prefix* / *prefix-length* [**eui-64**] **Example:** `Device(config-if)# ipv6 address 3ffe:b00:c18:1::3/127`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface.
Step 5	**tunnel source** {*ip-address* | *ipv6-address* | *interface-type interface-number*} **Example:** `Device(config-if)# tunnel source ethernet 0`	Specifies the source IPv4 address or the source interface type and number for the tunnel interface. • If an interface is specified, the interface must be configured with an IPv4 address.
Step 6	**tunnel destination** {*host-name* | *ip-address* | *ipv6-address*} **Example:** `Device(config-if)# tunnel destination 2001:DB8:1111:2222::1/64`	Specifies the destination IPv6 address or hostname for the tunnel interface.
Step 7	**tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre**| **gre multipoint** | **gre ipv6** | **ipip** [**decapsulate-any**] | **iptalk** | **ipv6** | **mpls** | **nos** **Example:** `Device(config-if)# tunnel mode gre ipv6`	Specifies a GRE IPv6 tunnel. **Note** The **tunnel mode gre ipv6** command specifies GRE as the encapsulation protocol for the tunnel.

Configuration Example: Tunnel Destination Address for IPv6 Tunnel

```
Device> enable
Device# configure terminal
Device(config)# interface Tunnel 0
Device(config-if)# ipv6 address 2001:1:1::1/48
Device(config-if)# tunnel source GigabitEthernet 0/0/0
Device(config-if)# tunnel destination 10.0.0.2
Device(config-if)# tunnel mode gre ipv6
Device(config-if)# exit
!
Device(config)# interface GigabitEthernet0/0/0
Device(config-if)# ip address 10.0.0.1 255.255.255.0
Device(config-if)# exit
!
Device(config)# ipv6 unicast-routing
Device(config)# router isis
Device(config-router)# net 49.0000.0000.000a.00
```

Additional References

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9600 Series Switches)*

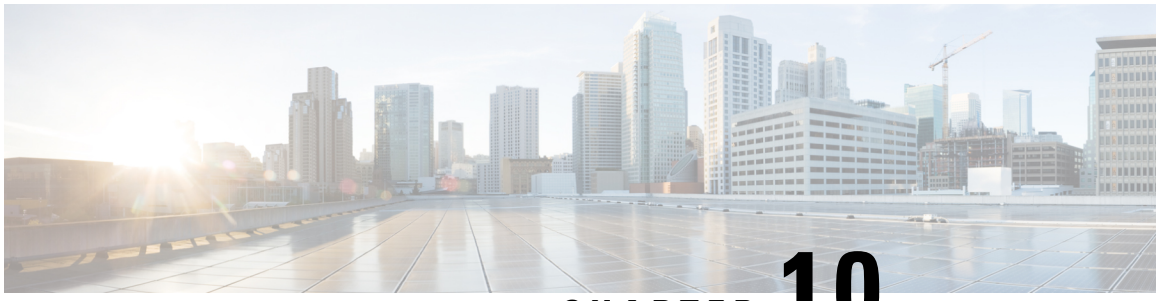
Feature History for IPv6 over IPv4 GRE Tunnels

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 Gibraltar 16.11.1	IPv6 over IPv4 GRE Tunnels	GRE tunnels are links between two points, with a separate tunnel for each link. The tunnels are not tied to a specific passenger or transport protocol, but in this case carry IPv6 as the passenger protocol with the GRE as the carrier protocol and IPv4 or IPv6 as the transport protocol.
Cisco IOS XE Cupertino 17.7.1	IPv6 over IPv4 GRE Tunnels	This feature was implemented on supervisor modules C9400X-SUP-2 and C9400X-SUP-2XL, which were introduced in this release.

Use the Cisco Feature Navigator to find information about platform and software image support.



CHAPTER 10

Configuring GLBP

- Restrictions for GLBP, on page 81
- Prerequisites for GLBP, on page 81
- Information About GLBP, on page 81
- How to Configure GLBP, on page 86
- Configuration Examples for GLBP, on page 97
- Additional References for GLBP, on page 98
- Feature History for GLBP, on page 98

Restrictions for GLBP

- Enhanced Object Tracking (EOT) is not stateful switchover (SSO)-aware and cannot be used with GLBP in SSO mode.

Prerequisites for GLBP

Before configuring GLBP, ensure that the devices can support multiple MAC addresses on the physical interfaces. For each GLBP forwarder to be configured, an additional MAC address is used.

Information About GLBP

GLBP Overview

GLBP provides automatic device backup for IP hosts configured with a single default gateway on an IEEE 802.3 LAN. Multiple first-hop devices on the LAN combine to offer a single virtual first-hop IP device while sharing the IP packet forwarding load. Other devices on the LAN act as redundant GLBP devices that will become active if any of the existing forwarding devices fail.

GLBP performs a similar function for the user as HSRP and VRRP. HSRP and VRRP allow multiple devices to participate in a virtual device group configured with a virtual IP address. One member is elected to be the active device to forward packets sent to the virtual IP address for the group. The other devices in the group are redundant until the active device fails. These standby devices have unused bandwidth that the protocol is

not using. Although multiple virtual device groups can be configured for the same set of devices, the hosts must be configured for different default gateways, which results in an extra administrative burden. The advantage of GLBP is that it additionally provides load balancing over multiple devices (gateways) using a single virtual IP address and multiple virtual MAC addresses. The forwarding load is shared among all devices in a GLBP group rather than being handled by a single device while the other devices stand idle. Each host is configured with the same virtual IP address, and all devices in the virtual device group participate in forwarding packets. GLBP members communicate between each other through hello messages sent every 3 seconds to the multicast address 224.0.0.102, UDP port 3222 (source and destination).

GLBP Packet Types

GLBP uses 3 different packet types to operate. The packet types are Hello, Request, and Reply. The Hello packet is used to advertise protocol information. Hello packets are multicast, and are sent when any virtual gateway or virtual forwarder is in Speak, Standby or Active state. Request and Reply packets are used for virtual MAC assignment. They are both unicast messages to and from the active virtual gateway (AVG).

GLBP Active Virtual Gateway

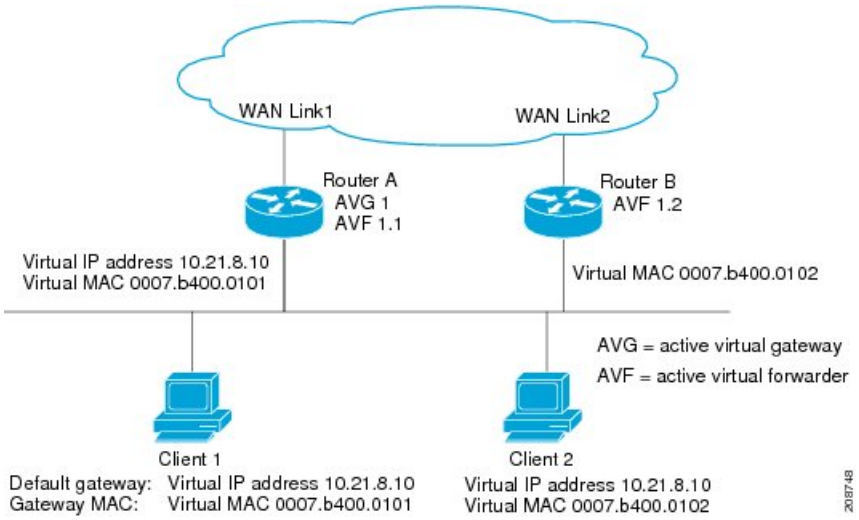
Members of a GLBP group elect one gateway to be the active virtual gateway (AVG) for that group. Other group members provide backup for the AVG if the AVG becomes unavailable. The AVG assigns a virtual MAC address to each member of the GLBP group. Each gateway assumes responsibility for forwarding packets sent to the virtual MAC address assigned to it by the AVG. These gateways are known as active virtual forwarders (AVFs) for their virtual MAC address.

The AVG is also responsible for answering Address Resolution Protocol(ARP) requests for the virtual IP address. Load sharing is achieved by the AVG replying to the ARP requests with different virtual MAC addresses.

When the **no glbp load-balancing**command is configured, if the AVG does not have an AVF, it preferentially responds to ARP requests with the MAC address of the first listening virtual forwarder (VF), which will causes traffic to route via another gateway until that VF migrates back to being the current AVG.

In the figure below, Router A (or Device A) is the AVG for a GLBP group, and is responsible for the virtual IP address 10.21.8.10. Router A is also an AVF for the virtual MAC address 0007.b400.0101. Router B (or Device B) is a member of the same GLBP group and is designated as the AVF for the virtual MAC address 0007.b400.0102. Client 1 has a default gateway IP address of 10.21.8.10 and a gateway MAC address of 0007.b400.0101. Client 2 shares the same default gateway IP address but receives the gateway MAC address 0007.b400.0102 because Router B is sharing the traffic load with Router A.

Figure 10: GLBP Topology



If Router A becomes unavailable, Client 1 will not lose access to the WAN because Router B will assume responsibility for forwarding packets sent to the virtual MAC address of Router A, and for responding to packets sent to its own virtual MAC address. Router B will also assume the role of the AVG for the entire GLBP group. Communication for the GLBP members continues despite the failure of a device in the GLBP group.

GLBP Virtual MAC Address Assignment

A GLBP group allows up to four virtual MAC addresses per group. The AVG is responsible for assigning the virtual MAC addresses to each member of the group. Other group members request a virtual MAC address after they discover the AVG through hello messages. Gateways are assigned the next MAC address in sequence. A virtual forwarder that is assigned a virtual MAC address by the AVG is known as a primary virtual forwarder. Other members of the GLBP group learn the virtual MAC addresses from hello messages. A virtual forwarder that has learned the virtual MAC address is referred to as a secondary virtual forwarder.

GLBP Virtual Gateway Redundancy

GLBP operates virtual gateway redundancy in the same way as HSRP. One gateway is elected as the AVG, another gateway is elected as the standby virtual gateway, and the remaining gateways are placed in a listen state.

If an AVG fails, the standby virtual gateway will assume responsibility for the virtual IP address. A new standby virtual gateway is then elected from the gateways in the listen state.

GLBP Virtual Forwarder Redundancy

Virtual forwarder redundancy is similar to virtual gateway redundancy with an AVF. If the AVF fails, one of the secondary virtual forwarders in the listen state assumes responsibility for the virtual MAC address.

The new AVF is also a primary virtual forwarder for a different forwarder number. GLBP migrates hosts away from the old forwarder number using two timers that start as soon as the gateway changes to the active virtual forwarder state. GLBP uses the hello messages to communicate the current state of the timers.

The redirect time is the interval during which the AVG continues to redirect hosts to the old virtual forwarder MAC address. When the redirect time expires, the AVG stops using the old virtual forwarder MAC address in ARP replies, although the virtual forwarder will continue to forward packets that were sent to the old virtual forwarder MAC address.

The secondary holdtime is the interval during which the virtual forwarder is valid. When the secondary holdtime expires, the virtual forwarder is removed from all gateways in the GLBP group. The expired virtual forwarder number becomes eligible for reassignment by the AVG.

GLBP Gateway Priority

GLBP gateway priority determines the role that each GLBP gateway plays and what happens if the AVG fails.

Priority also determines if a GLBP device functions as a backup virtual gateway and the order of ascendancy to becoming an AVG if the current AVG fails. You can configure the priority of each backup virtual gateway with a value of 1 through 255 using the **glbp priority** command.

In the "GLBP Topology" figure, if Router A (or Device A)—the AVG in a LAN topology—fails, an election process takes place to determine which backup virtual gateway should take over. In this example, Router B (or Device B) is the only other member in the group so it will automatically become the new AVG. If another device existed in the same GLBP group with a higher priority, then the device with the higher priority would be elected. If both devices have the same priority, the backup virtual gateway with the higher IP address would be elected to become the active virtual gateway.

By default, the GLBP virtual gateway preemptive scheme is disabled. A backup virtual gateway can become the AVG only if the current AVG fails, regardless of the priorities assigned to the virtual gateways. You can enable the GLBP virtual gateway preemptive scheme using the **glbp preempt** command. Preemption allows a backup virtual gateway to become the AVG, if the backup virtual gateway is assigned a higher priority than the current AVG.

GLBP Gateway Weighting and Tracking

GLBP uses a weighting scheme to determine the forwarding capacity of each device in the GLBP group. The weighting assigned to a device in the GLBP group can be used to determine whether it will forward packets and, if so, the proportion of hosts in the LAN for which it will forward packets. Thresholds can be set to disable forwarding when the weighting for a GLBP group falls below a certain value, and when it rises above another threshold, forwarding is automatically reenabled.

The GLBP group weighting can be automatically adjusted by tracking the state of an interface within the device. If a tracked interface goes down, the GLBP group weighting is reduced by a specified value. Different interfaces can be tracked to decrement the GLBP weighting by varying amounts.

By default, the GLBP virtual forwarder preemptive scheme is enabled with a delay of 30 seconds. A backup virtual forwarder can become the AVF if the current AVF weighting falls below the low weighting threshold for 30 seconds. You can disable the GLBP forwarder preemptive scheme using the **no glbp forwarder preempt** command or change the delay using the **glbp forwarder preempt delay minimum** command.

GLBP MD5 Authentication

GLBP MD5 authentication uses the industry-standard MD5 algorithm for improved reliability and security. MD5 authentication provides greater security than the alternative plain text authentication scheme and protects against spoofing software.

MD5 authentication allows each GLBP group member to use a secret key to generate a keyed MD5 hash that is part of the outgoing packet. A keyed hash of an incoming packet is generated and, if the hash within the incoming packet does not match the generated hash, the packet is ignored.

The key for the MD5 hash can either be given directly in the configuration using a key string or supplied indirectly through a key chain. The key string cannot exceed 100 characters in length.

A device will ignore incoming GLBP packets from devices that do not have the same authentication configuration for a GLBP group. GLBP has three authentication schemes:

- No authentication
- Plain text authentication
- MD5 authentication

GLBP packets will be rejected in any of the following cases:

- The authentication schemes differ on the device and in the incoming packet.
- MD5 digests differ on the device and in the incoming packet.
- Text authentication strings differ on the device and in the incoming packet.

ISSU-GLBP

This feature is supported on Cisco Catalyst 9600 Series Switches GLBP supports In Service Software Upgrade (ISSU). ISSU allows a high-availability (HA) system to run in Stateful Switchover (SSO) mode even when different versions of Cisco IOS software are running on the active and standby Route Processors (RPs) or line cards.

ISSU provides the ability to upgrade or downgrade from one supported Cisco IOS release to another while continuing to forward packets and maintain sessions, thereby reducing planned outage time. The ability to upgrade or downgrade is achieved by running different software versions on the active RP and standby RP for a short period of time to maintain state information between RPs. This feature allows the system to switch over to a secondary RP running upgraded (or downgraded) software and continue forwarding packets without session loss and with minimal or no packet loss. This feature is enabled by default.

GLBP SSO

This feature is supported on Cisco Catalyst 9600 Series Switches With the introduction of the GLBP SSO functionality, GLBP is stateful switchover (SSO) aware. GLBP can detect when a device is failing over to the secondary router processor (RP) and continue in its current group state.

SSO functions in networking devices (usually edge devices) that support dual RPs. SSO provides RP redundancy by establishing one of the RPs as the active processor and the other RP as the standby processor. SSO also synchronizes critical state information between the RPs so that network state information is dynamically maintained between RPs.

Without SSO-awareness, if GLBP is deployed on a device with redundant RPs, a switchover of roles between the active RP and the standby RP results in the device relinquishing its activity as a GLBP group member and then rejoining the group as if it had been reloaded. The GLBP SSO feature enables GLBP to continue its activities as a group member during a switchover. GLBP state information between redundant RPs is maintained so that the standby RP can continue the device's activities within the GLBP during and after a switchover.

This feature is enabled by default. To disable this feature, use the command **no glbp sso** in global configuration mode.

GLBP Benefits

Load Sharing

You can configure GLBP in such a way that traffic from LAN clients can be shared by multiple devices, thereby sharing the traffic load more equitably among available devices.

Multiple Virtual Devices

GLBP supports up to 1024 virtual devices (GLBP groups) on each physical interface of a device and up to four virtual forwarders per group.

Preemption

The redundancy scheme of GLBP enables you to preempt an active virtual gateway (AVG) with a higher priority backup virtual gateway that has become available. Forwarder preemption works in a similar way, except that forwarder preemption uses weighting instead of priority and is enabled by default.

Authentication

GLBP supports the industry-standard message digest 5 (MD5) algorithm for improved reliability, security, and protection against GLBP-spoofing software. A device within a GLBP group with a different authentication string than other devices will be ignored by other group members. You can alternatively use a simple text password authentication scheme between GLBP group members to detect configuration errors.

How to Configure GLBP

Customizing GLBP

Customizing the behavior of GLBP is optional. Be aware that as soon as you enable a GLBP group, that group is operating. It is possible that if you first enable a GLBP group before customizing GLBP, the device could take over control of the group and become the AVG before you have finished customizing the feature. Therefore, if you plan to customize GLBP, it is a good idea to do so before enabling GLBP.

Procedure

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/1`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**ip address** *ip-address mask* [**secondary**] **Example:** `Device(config-if)# ip address 10.21.8.32 255.255.255.0`	Specifies a primary or secondary IP address for an interface.
Step 5	**glbp** *group* **timers** [**msec**] *hellotime* [**msec**] *holdtime* **Example:** `Device(config-if)# glbp 10 timers 5 18`	Configures the interval between successive hello packets sent by the AVG in a GLBP group. • The *holdtime* argument specifies the interval in seconds before the virtual gateway and virtual forwarder information in the hello packet is considered invalid. • The optional **msec** keyword specifies that the following argument will be expressed in milliseconds, instead of the default seconds.
Step 6	**glbp** *group* **timers redirect** *redirect timeout* **Example:** `Device(config-if)# glbp 10 timers redirect 1800 28800`	Configures the time interval during which the AVG continues to redirect clients to an AVF. The default is 600 seconds (10 minutes). • The *timeout* argument specifies the interval in seconds before a secondary virtual forwarder becomes invalid. The default is 14,400 seconds (4 hours). **Note** The zero value for the *redirect* argument cannot be removed from the range of acceptable values because preexisting configurations of Cisco IOS software already using the zero value could be negatively

	Command or Action	Purpose
		affected during an upgrade. However, a zero setting is not recommended and, if used, results in a redirect timer that never expires. If the redirect timer does not expire, and the device fails, new hosts continue to be assigned to the failed device instead of being redirected to the backup.
Step 7	**glbp** *group* **load-balancing** [**host-dependent** | **round-robin** | **weighted**] **Example:** `Device(config-if)# glbp 10 load-balancing host-dependent`	Specifies the method of load balancing used by the GLBP AVG.
Step 8	**glbp** *group* **priority** *level* **Example:** `Device(config-if)# glbp 10 priority 254`	Sets the priority level of the gateway within a GLBP group. • The default value is 100.
Step 9	**glbp** *group* **preempt** [**delay minimum** *seconds*] **Example:** `Device(config-if)# glbp 10 preempt delay minimum 60`	Configures the device to take over as AVG for a GLBP group if it has a higher priority than the current AVG. • This command is disabled by default. • Use the optional **delay** and **minimum** keywords and the *seconds* argument to specify a minimum delay interval in seconds before preemption of the AVG takes place.
Step 10	**glbp** *group* **client-cache maximum** *number* [**timeout** *minutes*] **Example:** `Device(config-if)# glbp 10 client-cache maximum 1200 timeout 245`	(Optional) Enables the GLBP client cache. • This command is disabled by default. • Use the *number* argument to specify the maximum number of clients the cache will hold for this GLBP group. The range is from 8 to 2000. • Use the optional **timeout** *minutes* keyword and argument pair to configure the maximum amount of time a client entry can stay in the GLBP client cache after the client information was last updated. The range is from 1 to 1440 minutes (one day). **Note**

	Command or Action	Purpose
		For IPv4 networks, Cisco recommends setting a GLBP client cache timeout value that is slightly longer than the maximum expected end-host Address Resolution Protocol (ARP) cache timeout value.
Step 11	**glbp** *group* **name** *redundancy-name* **Example:** `Device(config-if)# glbp 10 name abc123`	Enables IP redundancy by assigning a name to the GLBP group. • The GLBP redundancy client must be configured with the same GLBP group name so the redundancy client and the GLBP group can be connected.
Step 12	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode, and returns the device to global configuration mode.
Step 13	**no glbp sso** **Example:** `Device(config)# no glbp sso`	(Optional) Disables GLBP support of SSO.

Configuring GLBP MD5 Authentication Using a Key String

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/1`	Configures an interface type and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ip address** *ip-address mask* [**secondary**] **Example:** `Device(config-if)# ip address 10.0.0.1 255.255.255.0`	Specifies a primary or secondary IP address for an interface.
Step 5	**glbp** *group-number* **authentication md5 key-string** [**0** | **7**] *key* **Example:** `Device(config-if)# glbp 1 authentication md5 key-string d00b4r987654321a`	Configures an authentication key for GLBP MD5 authentication. • The key string cannot exceed 100 characters in length. • No prefix to the *key* argument or specifying **0** means the key is unencrypted. • Specifying **7** means the key is encrypted. The key-string authentication key will automatically be encrypted if the **service password-encryption** global configuration command is enabled.
Step 6	**glbp** *group-number* **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# glbp 1 ip 10.0.0.10`	Enables GLBP on an interface and identifies the primary IP address of the virtual gateway.
Step 7	Repeat Steps 1 through 6 on each device that will communicate.	—
Step 8	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 9	**show glbp** **Example:** `Device# show glbp`	(Optional) Displays GLBP information. • Use this command to verify your configuration. The key string and authentication type will be displayed if configured.

Configuring GLBP MD5 Authentication Using a Key Chain

Perform this task to configure GLBP MD5 authentication using a key chain. Key chains allow a different key string to be used at different times according to the key chain configuration. GLBP will query the appropriate key chain to obtain the current live key and key ID for the specified key chain.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**key chain** *name-of-chain* **Example:** `Device(config)# key chain glbp2`	Enables authentication for routing protocols and identifies a group of authentication keys and enters key-chain configuration mode.
Step 4	**key** *key-id* **Example:** `Device(config-keychain)# key 100`	Identifies an authentication key on a key chain. • The value for the *key-id* argument must be a number.
Step 5	**key-string** *string* **Example:** `Device(config-keychain-key)# key-string abc123`	Specifies the authentication string for a key and enters key-chain key configuration mode. • The value for the *string* argument can be 1 to 80 uppercase or lowercase alphanumeric characters; the first character cannot be a numeral.
Step 6	**exit** **Example:** `Device(config-keychain-key)# exit`	Returns to key-chain configuration mode.
Step 7	**exit** **Example:** `Device(config-keychain)# exit`	Returns to global configuration mode.
Step 8	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/1`	Configures an interface type and enters interface configuration mode.

	Command or Action	Purpose
Step 9	**ip address** *ip-address mask* [**secondary**] **Example:** `Device(config-if)# ip address 10.21.0.1 255.255.255.0`	Specifies a primary or secondary IP address for an interface.
Step 10	**glbp** *group-number* **authentication md5 key-chain** *name-of-chain* **Example:** `Device(config-if)# glbp 1 authentication md5 key-chain glbp2`	Configures an authentication MD5 key chain for GLBP MD5 authentication. • The key chain name must match the name specified in Step 3.
Step 11	**glbp** *group-number* **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# glbp 1 ip 10.21.0.12`	Enables GLBP on an interface and identifies the primary IP address of the virtual gateway.
Step 12	Repeat Steps 1 through 10 on each device that will communicate.	—
Step 13	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 14	**show glbp** **Example:** `Device# show glbp`	(Optional) Displays GLBP information. • Use this command to verify your configuration. The key chain and authentication type will be displayed if configured.
Step 15	**show key chain** **Example:** `Device# show key chain`	(Optional) Displays authentication key information.

Configuring GLBP Text Authentication

Text authentication provides minimal security. Use MD5 authentication if security is required.

Procedure

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/1`	Configures an interface type and enters interface configuration mode.
Step 4	**ip address** *ip-address mask* [**secondary**] **Example:** `Device(config-if)# ip address 10.0.0.1 255.255.255.0`	Specifies a primary or secondary IP address for an interface.
Step 5	**glbp** *group-number* **authentication text** *string* **Example:** `Device(config-if)# glbp 10 authentication text stringxyz`	Authenticates GLBP packets received from other devices in the group. • If you configure authentication, all devices within the GLBP group must use the same authentication string.
Step 6	**glbp** *group-number* **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# glbp 1 ip 10.0.0.10`	Enables GLBP on an interface and identifies the primary IP address of the virtual gateway.
Step 7	Repeat Steps 1 through 6 on each device that will communicate.	—
Step 8	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 9	**show glbp** **Example:** `Device# show glbp`	(Optional) Displays GLBP information. • Use this command to verify your configuration.

Configuring GLBP Weighting Values and Object Tracking

GLBP weighting is used to determine whether a GLBP group can act as a virtual forwarder. Initial weighting values can be set and optional thresholds specified. Interface states can be tracked and a decrement value set to reduce the weighting value if the interface goes down. When the GLBP group weighting drops below a specified value, the group will no longer be an active virtual forwarder. When the weighting rises above a specified value, the group can resume its role as an active virtual forwarder.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *object-number* **interface** *type number* {**line-protocol** |{**ip** | **ipv6**} **routing**} **Example:** `Device(config)# track 2 interface GigabitEthernet 1/0/1 ip routing`	Configures an interface to be tracked where changes in the state of the interface affect the weighting of a GLBP gateway, and enters tracking configuration mode. • This command configures the interface and corresponding object number to be used with the **glbp weighting track** command. • The **line-protocol** keyword tracks whether the interface is up. The **ip routing** keywords also check that IP routing is enabled on the interface, and an IP address is configured.
Step 4	**exit** **Example:** `Device(config-track)# exit`	Returns to global configuration mode.
Step 5	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/1`	Enters interface configuration mode.

	Command or Action	Purpose
Step 6	**glbp** *group* **weighting** *maximum* [**lower** *lower*] [**upper** *upper*] **Example:** `Device(config-if)# glbp 10 weighting 110 lower 95 upper 105`	Specifies the initial weighting value, and the upper and lower thresholds, for a GLBP gateway.
Step 7	**glbp** *group* **weighting track** *object-number* [**decrement** *value*] **Example:** `Device(config-if)# glbp 10 weighting track 2 decrement 5`	Specifies an object to be tracked that affects the weighting of a GLBP gateway. • The *value* argument specifies a reduction in the weighting of a GLBP gateway when a tracked object fails.
Step 8	**glbp** *group* **forwarder preempt** [**delay minimum** *seconds*] **Example:** `Device(config-if)# glbp 10 forwarder preempt delay minimum 60`	Configures the device to take over as AVF for a GLBP group if the current AVF for a GLBP group falls below its low weighting threshold. • This command is enabled by default with a delay of 30 seconds. • Use the optional **delay** and **minimum** keywords and the *seconds* argument to specify a minimum delay interval in seconds before preemption of the AVF takes place.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Returns to privileged EXEC mode.
Step 10	**show track** [*object-number* | **brief**] [**interface** [**brief**] | **ip route** [**brief**] | **resolution** | **timers**] **Example:** `Device# show track 2`	Displays tracking information.

Troubleshooting GLBP

GLBP introduces five privileged EXEC mode commands to enable display of diagnostic output concerning various events relating to the operation of GLBP. The **debug condition glbp**,**debug glbp errors**, **debug glbp events**, **debug glbp packets**, and **debug glbp terse** commands are intended only for troubleshooting purposes because the volume of output generated by the software can result in severe performance degradation on the device. Perform this task to minimize the impact of using the **debug glbp** commands.

This procedure will minimize the load on the device created by the **debug condition glbp**or **debug glbp** command because the console port is no longer generating character-by-character processor interrupts. If you cannot connect to a console directly, you can run this procedure via a terminal server. If you must break the

Telnet connection, however, you may not be able to reconnect because the device may be unable to respond due to the processor load of generating the debugging output.

Before you begin

This task requires a device running GLBP to be attached directly to a console.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> 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 logging console** **Example:** `Device(config)# no logging console`	Disables all logging to the console terminal. • To reenable logging to the console, use the**logging console** command in global configuration mode.
Step 4	Use Telnet to access a device port and repeat Steps 1 and 2.	Enters global configuration mode in a recursive Telnet session, which allows the output to be redirected away from the console port.
Step 5	**end** **Example:** `Device(config)# end`	Exits to privileged EXEC mode.
Step 6	**terminal monitor** **Example:** `Device# terminal monitor`	Enables logging output on the virtual terminal.
Step 7	**debug condition glbp** *interface-type interface-number group* [*forwarder*] **Example:** `Device# debug condition glbp GigabitEthernet 0/0/0 1`	Displays debugging messages about GLBP conditions. • Try to enter only specific **debug condition glbp** or **debug glbp** commands to isolate the output to a certain subcomponent and minimize the load on the processor. Use appropriate arguments and keywords to generate more detailed debug information on specified subcomponents.

	Command or Action	Purpose
		• Enter the specific **no debug condition glbp** or **no debug glbp** command when you are finished.
Step 8	**terminal no monitor** **Example:** `Device# terminal no monitor`	Disables logging on the virtual terminal.

Configuration Examples for GLBP

Example: Customizing GLBP Configuration

```
Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# ip address 10.21.8.32 255.255.255.0
Device(config-if)# glbp 10 timers 5 18
Device(config-if)# glbp 10 timers redirect 1800 28800
Device(config-if)# glbp 10 load-balancing host-dependent
Device(config-if)# glbp 10 priority 254
Device(config-if)# glbp 10 preempt delay minimum 60

Device(config-if)# glbp 10 client-cache maximum 1200 timeout 245
```

Example: Configuring GLBP MD5 Authentication Using Key Strings

The following example shows how to configure GLBP MD5 authentication using a key string:

```
Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# ip address 10.0.0.1 255.255.255.0
Device(config-if)# glbp 2 authentication md5 key-string ThisStringIsTheSecretKey
Device(config-if)# glbp 2 ip 10.0.0.10
```

Example: Configuring GLBP MD5 Authentication Using Key Chains

In the following example, GLBP queries the key chain “AuthenticateGLBP” to obtain the current live key and key ID for the specified key chain:

```
Device(config)# key chain AuthenticateGLBP
Device(config-keychain)# key 1
Device(config-keychain-key)# key-string ThisIsASecretKey
Device(config-keychain-key)# exit
Device(config-keychain)# exit
Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# ip address 10.0.0.1 255.255.255.0
Device(config-if)# glbp 2 authentication md5 key-chain AuthenticateGLBP
Device(config-if)# glbp 2 ip 10.0.0.10
```

Example: Configuring GLBP Text Authentication

```
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# ip address 10.21.8.32 255.255.255.0
Device(config-if)# glbp 10 authentication text stringxyz
Device(config-if)# glbp 10 ip 10.21.8.10
```

Example: Configuring GLBP Weighting

In the following example, the device is configured to track the IP routing state of the POS interface 5/0/0 and 6/0/0, an initial GLBP weighting with upper and lower thresholds is set, and a weighting decrement value of 10 is set. If POS interface 5/0/0 and 6/0/0 go down, the weighting value of the device is reduced.

```
Device(config)# track 1 interface GigabitEthernet 1/0/1 line-protocol
Device(config)# track 2 interface GigabitEthernet 1/0/3 line-protocol
Device(config)# interface TenGigabitEthernet 0/0/1
Device(config-if)# ip address 10.21.8.32 255.255.255.0
Device(config-if)# glbp 10 weighting 110 lower 95 upper 105
Device(config-if)# glbp 10 weighting track 1 decrement 10
Device(config-if)# glbp 10 weighting track 2 decrement 10
```

Example: Enabling GLBP Configuration

In the following example, the device is configured to enable GLBP, and the virtual IP address of 10.21.8.10 is specified for GLBP group 10:

```
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# ip address 10.21.8.32 255.255.255.0
Device(config-if)# glbp 10 ip 10.21.8.10
```

Additional References for GLBP

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	See the IP Multicast Routing Commands section of the *Command Reference (Catalyst 9600 Series Switches)*

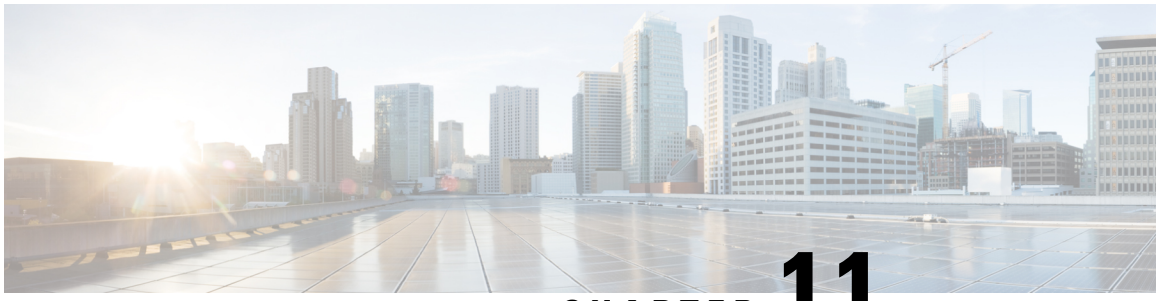
Feature History for GLBP

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 Gibraltar 16.12.1	Gateway Load Balancing Protocol	GLBP protects data traffic from a failed router or circuit, like HSRP and VRRP, while allowing packet load sharing between a group of redundant routers.
	GLBP MD5 Authentication	MD5 authentication provides greater security than the alternative plain text authentication scheme. MD5 authentication allows each GLBP group member to use a secret key to generate a keyed MD5 hash that is part of the outgoing packet. A keyed hash of an incoming packet is generated and, if the hash within the incoming packet does not match the generated hash, the packet is ignored.
	SSO—GLBP	GLBP is now SSO aware. GLBP can detect when a router is failing over to the secondary RP and continue in its current GLBP group state. Prior to being SSO aware, GLBP was not able to detect that a second RP was installed and configured to take over in the event that the primary RP failed. When the primary failed, the GLBP device would stop participating in the GLBP group and, depending on its role, could trigger another router in the group to take over as the active router. With this enhancement, GLBP detects the failover to the secondary RP and no change occurs to the GLBP group. If the secondary RP fails and the primary is still not available, then the GLBP group detects this and re-elects a new active GLBP router.
Cisco IOS XE Cupertino 17.7.1	Gateway Load Balancing Protocol GLBP MD5 Authentication SSO—GLBP	These features were implemented on supervisor modules C9400X-SUP-2 and C9400X-SUP-2XL, which were introduced in this release.

Use the Cisco Feature Navigator to find information about platform and software image support.



CHAPTER 11

Configuring HSRP

- Restriction About Hot Standby Router Protocol, on page 101
- Information About Hot Standby Router Protocol, on page 101
- How to Configure Hot Standby Router Protocol, on page 105
- Verifying HSRP Configurations, on page 121
- Configuration Examples for Hot Standby Router Protocol, on page 122
- Additional References for Configuring HSRP, on page 125
- Feature History for HSRP, on page 125

Restriction About Hot Standby Router Protocol

For Cisco Catalyst 9600 Series Supervisor 2 Module, Hot Standby Router Protocol (HSRP) is supported only on the switch virtual interface (SVI) and not on any other routed interfaces.

Information About Hot Standby Router Protocol

The following sections provide information about Hot Standby Router Protocol (HSRP)

HSRP Overview

HSRP is Cisco's standard method of providing high network availability by providing first-hop redundancy for IP hosts on an IEEE 802 LAN configured with a default gateway IP address. HSRP routes IP traffic without relying on the availability of any single router. It enables a set of router interfaces to work together to present the appearance of a single virtual router or default gateway to the hosts on a LAN. When HSRP is configured on a network or segment, it provides a virtual Media Access Control (MAC) address and an IP address that is shared among a group of configured routers. HSRP allows two or more HSRP-configured routers to use the MAC address and IP network address of a virtual router. The virtual router does not exist; it represents the common target for routers that are configured to provide backup to each other. One of the routers is selected to be the active router and another to be the standby router, which assumes control of the group MAC address and IP address should the designated active router fail.

Note Routers in an HSRP group can be any router interface that supports HSRP, including routed ports and switch virtual interfaces (SVIs).

HSRP provides high network availability by providing redundancy for IP traffic from hosts on networks. In a group of router interfaces, the active router is the router of choice for routing packets; the standby router is the router that takes over the routing duties when an active router fails or when preset conditions are met.

HSRP is useful for hosts that do not support a router discovery protocol and cannot switch to a new router when their selected router reloads or loses power. When HSRP is configured on a network segment, it provides a virtual MAC address and an IP address that is shared among router interfaces in a group of router interfaces running HSRP. The router selected by the protocol to be the active router receives and routes packets destined for the group's MAC address. For n routers running HSRP, there are n +1 IP and MAC addresses assigned.

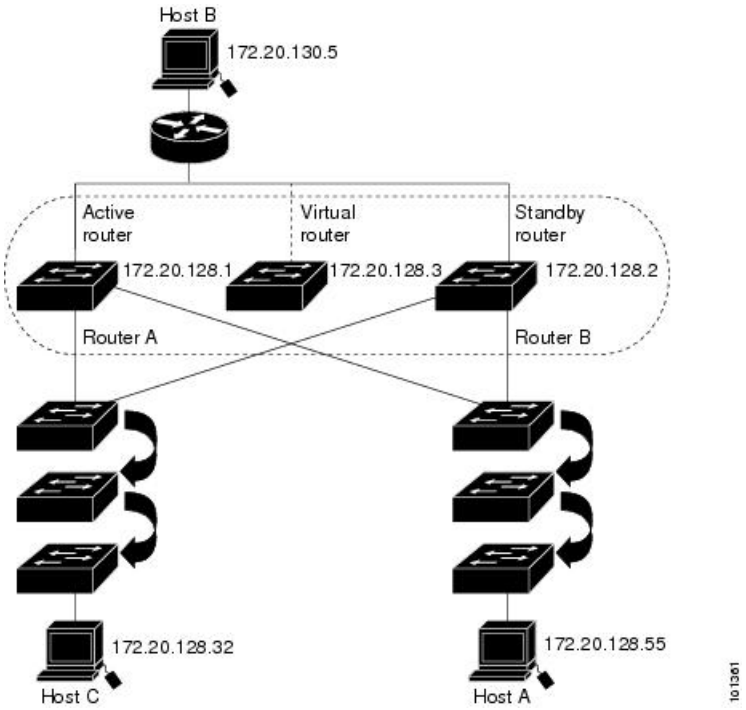
HSRP detects when the designated active router fails, and a selected standby router assumes control of the Hot Standby group's MAC and IP addresses. A new standby router is also selected at that time. Devices running HSRP send and receive multicast UDP-based hello packets to detect router failure and to designate active and standby routers. When HSRP is configured on an interface, Internet Control Message Protocol (ICMP) redirect messages are automatically enabled for the interface.

You can configure multiple Hot Standby groups among switches and switch stacks that are operating in Layer 3 to make more use of the redundant routers.

To do so, specify a group number for each Hot Standby command group you configure for an interface. For example, you might configure an interface on switch 1 as an active router and one on switch 2 as a standby router and also configure another interface on switch 2 as an active router with another interface on switch 1 as its standby router.

The following figure shows a segment of a network configured for HSRP. Each router is configured with the MAC address and IP network address of the virtual router. Instead of configuring hosts on the network with the IP address of Router A, you configure them with the IP address of the virtual router as their default router. When Host C sends packets to Host B, it sends them to the MAC address of the virtual router. If for any reason, Router A stops transferring packets, Router B responds to the virtual IP address and virtual MAC address and becomes the active router, assuming the active router duties. Host C continues to use the IP address of the virtual router to address packets destined for Host B, which Router B now receives and sends to Host B. Until Router A resumes operation, HSRP allows Router B to provide uninterrupted service to users on Host C's segment that need to communicate with users on Host B's segment and also continues to perform its normal function of handling packets between the Host A segment and Host B.

Figure 11: Typical HSRP Configuration



HSRP Versions

Cisco IOS XE Gibraltar 16.11.1 and later support these Hot Standby Router Protocol (HSRP) versions:

The switch supports these HSRP versions:

- HSRPv1- Version 1 of the HSRP, the default version of HSRP. It has these features:
  - The HSRP group number can be from 0 to 255.
  - HSRPv1 uses the multicast address 224.0.0.2 to send hello packets, which can conflict with Cisco Group Management Protocol (CGMP) leave processing. You cannot enable HSRPv1 and CGMP at the same time; they are mutually exclusive.
- HSRPv2- Version 2 of the HSRP has these features:
  - HSRPv2 uses the multicast address 224.0.0.102 to send hello packets. HSRPv2 and CGMP leave processing are no longer mutually exclusive, and both can be enabled at the same time.
  - HSRPv2 has a different packet format than HRSPv1.

A switch running HSRPv1 cannot identify the physical router that sent a hello packet because the source MAC address of the router is the virtual MAC address.

HSRPv2 has a different packet format than HSRPv1. A HSRPv2 packet uses the type-length-value (TLV) format and has a 6-byte identifier field with the MAC address of the physical router that sent the packet.

If an interface running HSRPv1 gets an HSRPv2 packet, the type field is ignored.

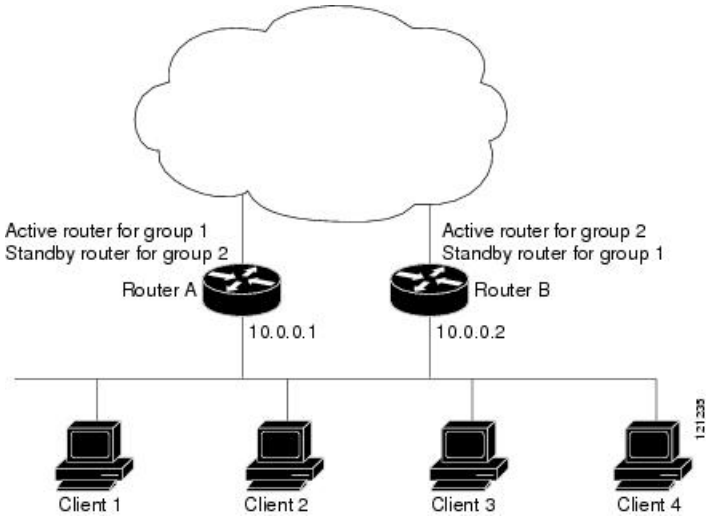
Multiple HSRP

The switch supports Multiple HSRP (MHSRP), an extension of HSRP that allows load sharing between two or more HSRP groups. You can configure MHSRP to achieve load-balancing and to use two or more standby groups (and paths) from a host network to a server network.

In the figure below, half the clients are configured for Router A, and half the clients are configured for Router B. Together, the configuration for Routers A and B establishes two HSRP groups. For group 1, Router A is the default active router because it has the assigned highest priority, and Router B is the standby router. For group 2, Router B is the default active router because it has the assigned highest priority, and Router A is the standby router. During normal operation, the two routers share the IP traffic load. When either router becomes unavailable, the other router becomes active and assumes the packet-transfer functions of the router that is unavailable.

Note For MHSRP, you need to enter the **standby preempt** interface configuration command on the HSRP interfaces so that if a router fails and then comes back up, preemption restores load sharing.

Figure 12: MHSRP Load Sharing



SSO HSRP

SSO HSRP alters the behavior of HSRP when a device with redundant Route Processors (RPs) is configured for stateful switchover (SSO) redundancy mode. When an RP is active and the other RP is standby, SSO enables the standby RP to take over if the active RP fails.

With this functionality, HSRP SSO information is synchronized to the standby RP, allowing traffic that is sent using the HSRP virtual IP address to be continuously forwarded during a switchover without a loss of data or a path change. Additionally, if both RPs fail on the active HSRP device, then the standby HSRP device takes over as the active HSRP device.

The feature is enabled by default when the redundancy mode of operation is set to SSO.

HSRP and Switch Stacks

HSRP hello messages are generated by the active switch. If HSRP fails on the active switch, a flap in the HSRP active state might occur. This is because HSRP hello messages are not generated while a new active switch is elected and initialized, and the standby router might become active after the active switch fails.

Configuring HSRP for IPv6

Switches running the Network Advantage license support the Hot Standby Router Protocol (HSRP) for IPv6. HSRP provides routing redundancy for routing IPv6 traffic not dependent on the availability of any single router. IPv6 hosts learn of available routers through IPv6 neighbor discovery router advertisement messages. These messages are multicast periodically or are solicited by hosts.

An HSRP IPv6 group has a virtual MAC address that is derived from the HSRP group number and a virtual IPv6 link-local address that is, by default, derived from the HSRP virtual MAC address.

Periodic messages are sent for the HSRP virtual IPv6 link-local address when the HSRP group is active. These messages stop after a final one is sent when the group leaves the active state.

Note When configuring HSRP for IPv6, you must enable HSRP version 2 (HSRPv2) on the interface.

HSRP IPv6 Virtual MAC Address Range

HSRP IPv6 uses a different virtual MAC address block than does HSRP for IP:

0005.73A0.0000 through 0005.73A0.0FFF (4096 addresses)

HSRP IPv6 UDP Port Number

Port number 2029 has been assigned to HSRP IPv6.

How to Configure Hot Standby Router Protocol

The following sections provide configuration information about HSRP.

Default HSRP Configuration

Table 5: Default HSRP Configuration

Feature	Default Setting
HSRP version	Version 1
HSRP groups	None configured
Standby group number	0
Standby MAC address	System assigned as: 0000.0c07.acXX, where XX is the HSRP group number

Feature	Default Setting
Standby priority	100
Standby delay	0 (no delay)
Standby track interface priority	10
Standby hello time	3 seconds
Standby holdtime	10 seconds

HSRP Configuration Guidelines

- HSRPv2 and HSRPv1 are mutually exclusive. HSRPv2 is not interoperable with HSRPv1 on an interface and the reverse.
- In the procedures, the specified interface must be one of these Layer 3 interfaces:
  - Routed port: A physical port configured as a Layer 3 port by entering the **no switchport** command in interface configuration mode.
  - SVI: A VLAN interface created by using the **interface vlan** *vlan_id* in global configuration mode, and by default a Layer 3 interface.
  - Etherchannel port channel in Layer 3 mode: A port-channel logical interface created by using the **interface port-channel** *port-channel-number* in global configuration mode, and binding the Ethernet interface into the channel group.
- All Layer 3 interfaces must have IP addresses assigned to them.
- HSRP millisecond timers are not supported.

Enabling HSRP

The **standby ip** interface configuration command activates HSRP on the configured interface. If an IP address is specified, that address is used as the designated address for the Hot Standby group. If no IP address is specified, the address is learned through the standby function. You must configure at least one Layer 3 port on the LAN with the designated address. Configuring an IP address always overrides another designated address currently in use.

When the **standby ip** command is enabled on an interface and proxy ARP is enabled, if the interface's Hot Standby state is active, proxy ARP requests are answered using the Hot Standby group MAC address. If the interface is in a different state, proxy ARP responses are suppressed.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device(config)# configure terminal`	
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/0/1`	Enters interface configuration mode, and enter the Layer 3 interface on which you want to enable HSRP.
Step 3	**standby version** {**1** | **2**} **Example:** `Device(config-if)# standby version 1`	(Optional) Configures the HSRP version on the interface. • 1- Selects HSRPv1. • 2- Selects HSRPv2. If you do not enter this command or do not specify a keyword, the interface runs the default HSRP version, HSRP v1.
Step 4	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 1 ip`	Creates (or enable) the HSRP group using its number and virtual IP address. • (Optional) group-number- The group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) ip-address- The virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**- The IP address is a secondary hot standby router interface. If neither router is designated as a secondary or standby router and no priorities are set, the primary IP addresses are compared and the higher IP address is the active router, with the next highest as the standby router.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode

	Command or Action	Purpose
Step 6	**show standby** [*interface-id* [*group*]] **Example:** `Device# show standby`	Verifies the configuration of the standby groups.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Enabling and Verifying an HSRP Group for IPv6 Operation

In this task, when you enter the **standby ipv6** command, a link-local address is generated from the link-local prefix, and a modified EUI-64 format interface identifier is generated in which the EUI-64 interface identifier is created from the relevant HSRP virtual MAC address.

A link-local address is an IPv6 unicast address that can be automatically configured on any interface using the link-local prefix FE80::/10 (1111 1110 10) and the interface identifier in the modified EUI-64 format. Link-local addresses are used in the stateless autoconfiguration process. Nodes on a local link can use link-local addresses to communicate; the nodes do not need site-local or globally unique addresses to communicate.

In IPv6, a device on the link advertises in RA messages any site-local and global prefixes, and its willingness to function as a default device for the link. RA messages are sent periodically and in response to router solicitation messages, which are sent by hosts at system startup.

A node on the link can automatically configure site-local and global IPv6 addresses by appending its interface identifier (64 bits) to the prefixes (64 bits) included in the RA messages. The resulting 128-bit IPv6 addresses configured by the node are then subjected to duplicate address detection to ensure their uniqueness on the link. If the prefixes advertised in the RA messages are globally unique, then the IPv6 addresses configured by the node are also guaranteed to be globally unique. Router solicitation messages, which have a value of 133 in the Type field of the ICMP packet header, are sent by hosts at system startup so that the host can immediately autoconfigure without needing to wait for the next scheduled RA message.

To enabling and verifying an HSRP group for IPv6, perform this procedure:

Procedure

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

	Command or Action	Purpose
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams. • The **ipv6 unicast-routing** command must be enabled for HSRP for IPv6 to work.
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 5	**standby** [*group-number*] **ipv6** {*link-local-address* | **autoconfig**} **Example:** `Device(config-if)# standby 1 ipv6 autoconfig`	Activates the HSRP in IPv6.
Step 6	**standby** [*group-number*] **preempt** [**delay minimum** *seconds* | **reload** *seconds* | **sync** *seconds*] **Example:** `Device(config-if)# standby 1 preempt`	Configures HSRP preemption and preemption delay.
Step 7	**standby** [*group-number*] **priority** *priority* **Example:** `Device(config-if)# standby 1 priority 110`	Configures HSRP priority.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Returns the device to privileged EXEC mode.
Step 9	**show standby** [*type number* [*group*]] [**all** | **brief**] **Example:** `Device# show standby`	Displays HSRP information.
Step 10	**show ipv6 interface** [**brief**] [*interface-type interface-number*] [**prefix**] **Example:**	Displays the usability status of interfaces configured for IPv6.

	Command or Action	Purpose
	`Device# show ipv6 interface GigabitEthernet 0/0/0`	

Configuring HSRP Priority

The **standby priority**, **standby preempt**, and **standby track** interface configuration commands are all used to set characteristics for finding active and standby routers and behavior regarding when a new active router takes over.

When configuring HSRP priority, follow these guidelines:

- Assigning a priority allows you to select the active and standby routers. If preemption is enabled, the router with the highest priority becomes the active router. If priorities are equal, the current active router does not change.
- The highest number (1 to 255) represents the highest priority (most likely to become the active router).
- When setting the priority, preempt, or both, you must specify at least one keyword (**priority**, **preempt**, or both)
- The priority of the device can change dynamically if an interface is configured with the **standby track** command and another interface on the router goes down.
- The **standby track** interface configuration command ties the router hot standby priority to the availability of its interfaces and is useful for tracking interfaces that are not configured for HSRP. When a tracked interface fails, the hot standby priority on the device on which tracking has been configured decreases by 10. If an interface is not tracked, its state changes do not affect the hot standby priority of the configured device. For each interface configured for hot standby, you can configure a separate list of interfaces to be tracked.
- The **standby track** *interface-priority* interface configuration command specifies how much to decrement the hot standby priority when a tracked interface goes down. When the interface comes back up, the priority is incremented by the same amount.
- When multiple tracked interfaces are down and *interface-priority* values have been configured, the configured priority decrements are cumulative. If tracked interfaces that were not configured with priority values fail, the default decrement is 10, and it is noncumulative.
- When routing is first enabled for the interface, it does not have a complete routing table. If it is configured to preempt, it becomes the active router, even though it is unable to provide adequate routing services. To solve this problem, configure a delay time to allow the router to update its routing table.

Beginning in privileged EXEC mode, use one or more of these steps to configure HSRP priority characteristics on an interface:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet1/0/1`	Enters interface configuration mode, and enter the HSRP interface on which you want to set priority.
Step 3	**standby** [*group-number*] **priority** *priority* **Example:** `Device(config-if)# standby 120 priority 50`	Sets a **priority** value used in choosing the active router. The range is 1 to 255; the default priority is 100. The highest number represents the highest priority. (Optional) group-number—The group number to which the command applies. Use the **no** form of the command to restore the default values.
Step 4	**standby** [*group-number*] **preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:** `Device(config-if)# standby 1 preempt delay 300`	Configures the router to **preempt**, which means that when the local router has a higher priority than the active router, it becomes the active router. • (Optional) group-number-The group number to which the command applies. • (Optional) **delay minimum**—Set to cause the local router to postpone taking over the active role for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). • (Optional) **delay reload**—Set to cause the local router to postpone taking over the active role after a reload for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over after a reload). • (Optional) **delay sync**—Set to cause the local router to postpone taking over the active role so that IP redundancy clients can reply (either with an ok or wait reply) for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). Use the **no** form of the command to restore the default values.

	Command or Action	Purpose
Step 5	**standby** [*group-number*] **track** *type number* [*interface-priority*] **Example:** `Device(config-if)# standby track interface gigabitethernet1/1/1`	Configures an interface to track other interfaces so that if one of the other interfaces goes down, the device's Hot Standby priority is lowered. • (Optional) group-number- The group number to which the command applies. • type- Enter the interface type (combined with interface number) that is tracked. • number- Enter the interface number (combined with interface type) that is tracked. • (Optional) interface-priority- Enter the amount by which the hot standby priority for the router is decremented or incremented when the interface goes down or comes back up. The default value is 10.
Step 6	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config**	Verifies the configuration of the standby groups.
Step 8	**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Configuring MHSRP

To enable MHSRP and load-balancing, you configure two routers as active routers for their groups, with virtual routers as standby routers as shown in the *MHSRP Load Sharing* figure in the Multiple HSRP section. You need to enter the **standby preempt** interface configuration command on each HSRP interface so that if a router fails and comes back up, the preemption occurs and restores load-balancing.

Router A is configured as the active router for group 1, and Router B is configured as the active router for group 2. The HSRP interface for Router A has an IP address of 10.0.0.1 with a group 1 standby priority of 110 (the default is 100). The HSRP interface for Router B has an IP address of 10.0.0.2 with a group 2 standby priority of 110.

Group 1 uses a virtual IP address of 10.0.0.3 and group 2 uses a virtual IP address of 10.0.0.4.

Configuring Router A

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet1/0/1`	Configures an interface type and enters interface configuration mode.
Step 3	**no switchport** **Example:** `Device(config)# no switchport`	Switches an interface that is in Layer 2 mode into Layer 3 mode for Layer 3 configuration.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.0.0.1 255.255.255.0`	Specifies an IP address for an interface.
Step 5	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 1 ip 10.0.0.3`	Creates the HSRP group using its number and virtual IP address. • (Optional) *group-number*- The group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) *ip-address*- The virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**- The IP address is a secondary hot standby router interface. If neither router is designated as a secondary or standby router and no priorities are set, the primary IP addresses are compared and the higher IP address

	Command or Action	Purpose
		is the active router, with the next highest as the standby router.
Step 6	**standby** [*group-number*] **priority** *priority* **Example:** `Device(config-if)# standby 1 priority 110`	Sets a **priority** value used in choosing the active router. The range is 1 to 255; the default priority is 100. The highest number represents the highest priority. (Optional) *group-number*—The group number to which the command applies. Use the **no** form of the command to restore the default values.
Step 7	**standby** [*group-number*] **preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:** `Device(config-if)# standby 1 preempt delay 300`	Configures the router to **preempt**, which means that when the local router has a higher priority than the active router, it becomes the active router. • (Optional) group-number-The group number to which the command applies. • (Optional) **delay minimum**—Set to cause the local router to postpone taking over the active role for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). • (Optional) **delay reload**—Set to cause the local router to postpone taking over the active role after a reload for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over after a reload) . • (Optional) **delay sync**—Set to cause the local router to postpone taking over the active role so that IP redundancy clients can reply (either with an ok or wait reply) for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). Use the **no** form of the command to restore the default values.

	Command or Action	Purpose
Step 8	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 2 ip 10.0.0.4`	Creates the HSRP group using its number and virtual IP address. • (Optional) *group-number*- The group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) *ip-address*- The virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**- The IP address is a secondary hot standby router interface. If neither router is designated as a secondary or standby router and no priorities are set, the primary IP addresses are compared and the higher IP address is the active router, with the next highest as the standby router.
Step 9	**standby** [*group-number*] **preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:** `Device(config-if)# standby 2 preempt delay 300`	Configures the router to **preempt**, which means that when the local router has a higher priority than the active router, it becomes the active router. • (Optional) group-number-The group number to which the command applies. • (Optional) **delay minimum**—Set to cause the local router to postpone taking over the active role for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). • (Optional) **delay reload**—Set to cause the local router to postpone taking over the active role after a reload for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over after a reload). • (Optional) **delay sync**—Set to cause the local router to postpone taking over the active role so that IP redundancy clients

	Command or Action	Purpose
		can reply (either with an ok or wait reply) for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). Use the **no** form of the command to restore the default values.
Step 10	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 11	**show running-config**	Verifies the configuration of the standby groups.
Step 12	**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Configuring Router B

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet1/0/1`	Configures an interface type and enters interface configuration mode.
Step 3	**no switchport** **Example:** `Device(config)# no switchport`	Switches an interface that is in Layer 2 mode into Layer 3 mode for Layer 3 configuration.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.0.0.2 255.255.255.0`	Specifies an IP address for an interface.

	Command or Action	Purpose
Step 5	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 1 ip 10.0.0.3`	Creates the HSRP group using its number and virtual IP address. • (Optional) *group-number*- The group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) *ip-address*- The virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**- The IP address is a secondary hot standby router interface. If neither router is designated as a secondary or standby router and no priorities are set, the primary IP addresses are compared and the higher IP address is the active router, with the next highest as the standby router.
Step 6	**standby** [*group-number*] **priority** *priority* **Example:** `Device(config-if)# standby 2 priority 110`	Sets a **priority** value used in choosing the active router. The range is 1 to 255; the default priority is 100. The highest number represents the highest priority. (Optional) *group-number*—The group number to which the command applies. Use the **no** form of the command to restore the default values.
Step 7	**standby** [*group-number*] **preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:** `Device(config-if)# standby 1 preempt delay 300`	Configures the router to **preempt**, which means that when the local router has a higher priority than the active router, it becomes the active router. • (Optional) group-number-The group number to which the command applies. • (Optional) **delay minimum**—Set to cause the local router to postpone taking over the active role for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over).

	Command or Action	Purpose
		• (Optional) **delay reload**—Set to cause the local router to postpone taking over the active role after a reload for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over after a reload). • (Optional) **delay sync**—Set to cause the local router to postpone taking over the active role so that IP redundancy clients can reply (either with an ok or wait reply) for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). Use the **no** form of the command to restore the default values.
Step 8	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 2 ip 10.0.0.4`	Creates the HSRP group using its number and virtual IP address. • (Optional) *group-number*- The group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) *ip-address*- The virtual IP address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**- The IP address is a secondary hot standby router interface. If neither router is designated as a secondary or standby router and no priorities are set, the primary IP addresses are compared and the higher IP address is the active router, with the next highest as the standby router.
Step 9	**standby** [*group-number*] **preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:**	Configures the router to **preempt**, which means that when the local router has a higher priority than the active router, it becomes the active router. • (Optional) group-number-The group number to which the command applies.

	Command or Action	Purpose
	`Device(config-if)# standby 2 preempt delay 300`	• (Optional) **delay minimum**—Set to cause the local router to postpone taking over the active role for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over) . • (Optional) **delay reload**—Set to cause the local router to postpone taking over the active role after a reload for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over after a reload). • (Optional) **delay sync**—Set to cause the local router to postpone taking over the active role so that IP redundancy clients can reply (either with an ok or wait reply) for the number of seconds shown. The range is 0 to 3600 seconds (1 hour); the default is 0 (no delay before taking over). Use the **no** form of the command to restore the default values.
Step 10	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 11	**show running-config**	Verifies the configuration of the standby groups.
Step 12	**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Configuring HSRP Authentication and Timers

You can optionally configure an HSRP authentication string or change the hello-time interval and hold-time interval.

When configuring these attributes, follow these guidelines:

- The authentication string is sent unencrypted in all HSRP messages. You must configure the same authentication string on all routers and access servers on a cable to ensure interoperation. Authentication mismatch prevents a device from learning the designated Hot Standby IP address and timer values from other routers configured with HSRP.

- Routers or access servers on which standby timer values are not configured can learn timer values from the active or standby router. The timers configured on an active router always override any other timer settings.
- All routers in a Hot Standby group should use the same timer values. Normally, the *holdtime* is greater than or equal to 3 times the *hellotime*.

Beginning in privileged EXEC mode, use one or more of these steps to configure HSRP authentication and timers on an interface:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config) # interface gigabitethernet1/0/1`	Enters interface configuration mode, and enter the HSRP interface on which you want to set priority.
Step 3	**standby** [*group-number*] **authentication** *string* **Example:** `Device(config-if) # standby 1 authentication word`	(Optional) **authentication** *string*—Enter a string to be carried in all HSRP messages. The authentication string can be up to eight characters in length; the default string is **cisco**. (Optional) *group-number*—The group number to which the command applies.
Step 4	**standby** [*group-number*] **timers** *hellotime holdtime* **Example:** `Device(config-if) # standby 1 timers 5 15`	(Optional) Configure the time interval to send and receive hello packets. • *group-number*—The group number to which the command applies. • *hellotime* —Set the interval between successive hello packets in seconds. The range is 1 to 255 seconds. The default is 3. • *holdtime*—Set the interval to wait for a hello packet from a neighbor device before declaring the neighbor device as inactive. The range is 1 to 255 seconds. The default is 10.
Step 5	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	
Step 6	**show running-config**	Verifies the configuration of the standby groups.
Step 7	**copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Enabling HSRP Support for ICMP Redirect Messages

ICMP redirect messages are automatically enabled on interfaces configured with HSRP. ICMP is a network layer Internet protocol that provides message packets to report errors and other information relevant to IP processing. ICMP provides diagnostic functions, such as sending and directing error packets to the host. This feature filters outgoing ICMP redirect messages through HSRP, in which the next hop IP address might be changed to an HSRP virtual IP address. For more information, see the Cisco IOS IP Configuration Guide, Release 12.4.

Configuring HSRP Groups and Clustering

When a device is participating in an HSRP standby routing and clustering is enabled, you can use the same standby group for command switch redundancy and HSRP redundancy. Use the **cluster standby-group** *HSRP-group-name* [**routing-redundancy**] global configuration command to enable the same HSRP standby group to be used for command switch and routing redundancy. If you create a cluster with the same HSRP standby group name without entering the **routing-redundancy** keyword, HSRP standby routing is disabled for the group.

Verifying HSRP Configurations

From privileged EXEC mode, use this command to display HSRP settings:

show standby [*interface-id* [*group*]] [**brief**] [**detail**]

You can display HSRP information for the whole switch, for a specific interface, for an HSRP group, or for an HSRP group on an interface. You can also specify whether to display a concise overview of HSRP information or detailed HSRP information. The default display is **detail**. If there are a large number of HSRP groups, using the **show standby** command without qualifiers can result in an unwieldy display.

Example

```
Switch #show standby
VLAN1 - Group 1
Local state is Standby, priority 105, may preempt
Hellotime 3 holdtime 10
Next hello sent in 00:00:02.182
Hot standby IP address is 172.20.128.3 configured
Active router is 172.20.128.1 expires in 00:00:09
Standby router is local
Standby virtual mac address is 0000.0c07.ac01
Name is bbb

VLAN1 - Group 100
```

```
Local state is Standby, priority 105, may preempt
Hellotime 3 holdtime 10
Next hello sent in 00:00:02.262
Hot standby IP address is 172.20.138.51 configured
Active router is 172.20.128.1 expires in 00:00:09
Active router is local
Standby router is unknown expired
Standby virtual mac address is 0000.0c07.ac64
Name is test
```

Configuration Examples for Hot Standby Router Protocol

The following sections provide various configuration examples for HSRP.

Enabling HSRP: Example

This example shows how to activate HSRP for group 1 on an interface. The IP address used by the hot standby group is learned by using HSRP.

Note This procedure is the minimum number of steps required to enable HSRP. Other configurations are optional.

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/0/1
Switch(config-if)# no switchport
Switch(config-if)# standby 1 ip
Switch(config-if)# end
Switch # show standby
```

Example: Configuration and Verification for an HSRP Group

The following example shows configuration and verification for an HSRP group for IPv6 that consists of Device1 and Device2. The **show standby** command is issued for each device to verify the device's configuration:

Device 1 configuration

```
interface FastEthernet0/0.100
description DATA VLAN for PCs
encapsulation dot1Q 100
ipv6 address 2001:DB8:CAFE:2100::BAD1:1010/64
standby version 2
standby 101 priority 120
standby 101 preempt delay minimum 30
standby 101 authentication ese
standby 101 track Serial0/1/0.17 90
standby 201 ipv6 autoconfig
standby 201 priority 120
standby 201 preempt delay minimum 30
standby 201 authentication ese
standby 201 track Serial0/1/0.17 90
Device1# show standby
FastEthernet0/0.100 - Group 101 (version 2)
```

```
State is Active
2 state changes, last state change 5w5d
Active virtual MAC address is 0000.0c9f.f065
Local virtual MAC address is 0000.0c9f.f065 (v2 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 2.296 secs
Authentication text "ese"
Preemption enabled, delay min 30 secs
Active router is local
Priority 120 (configured 120)
Track interface Serial0/1/0.17 state Up decrement 90
IP redundancy name is "hsrp-Fa0/0.100-101" (default)
FastEthernet0/0.100 - Group 201 (version 2)
State is Active
2 state changes, last state change 5w5d
Virtual IP address is FE80::5:73FF:FEA0:C9
Active virtual MAC address is 0005.73a0.00c9
Local virtual MAC address is 0005.73a0.00c9 (v2 IPv6 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 2.428 secs
Authentication text "ese"
Preemption enabled, delay min 30 secs
Active router is local
Standby router is FE80::20F:8FFF:FE37:3B70, priority 100 (expires in 7.856 sec)
Priority 120 (configured 120)
Track interface Serial0/1/0.17 state Up decrement 90
IP redundancy name is "hsrp-Fa0/0.100-201" (default)
```

Device 2 configuration

```
interface FastEthernet0/0.100
description DATA VLAN for Computers
encapsulation dot1Q 100
ipv6 address 2001:DB8:CAFE:2100::BAD1:1020/64
standby version 2
standby 101 preempt
standby 101 authentication ese
standby 201 ipv6 autoconfig
standby 201 preempt
standby 201 authentication ese
Device2# show standby
FastEthernet0/0.100 - Group 101 (version 2)
State is Standby
7 state changes, last state change 5w5d
Active virtual MAC address is 0000.0c9f.f065
Local virtual MAC address is 0000.0c9f.f065 (v2 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 0.936 secs
Authentication text "ese"
Preemption enabled
MAC address is 0012.7fc6.8f0c
Standby router is local
Priority 100 (default 100)
IP redundancy name is "hsrp-Fa0/0.100-101" (default)
FastEthernet0/0.100 - Group 201 (version 2)
State is Standby
7 state changes, last state change 5w5d
Virtual IP address is FE80::5:73FF:FEA0:C9
Active virtual MAC address is 0005.73a0.00c9
Local virtual MAC address is 0005.73a0.00c9 (v2 IPv6 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 0.936 secs
Authentication text "ese"
Preemption enabled
Active router is FE80::212:7FFF:FEC6:8F0C, priority 120 (expires in 7.548 sec)
```

```
MAC address is 0012.7fc6.8f0c
Standby router is local
Priority 100 (default 100)
IP redundancy name is "hsrp-Fa0/0.100-201" (default)
```

Configuring HSRP Priority: Example

This example activates a port, sets an IP address and a priority of 120 (higher than the default value), and waits for 300 seconds (5 minutes) before attempting to become the active router:

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/0/1
Switch(config-if)# no switchport
Switch(config-if)# standby ip 172.20.128.3
Switch(config-if)# standby priority 120 preempt delay 300
Switch(config-if)# end
Switch # show standby
```

Configuring MHSRP: Example

This example shows how to enable the MHSRP configuration shown in the figure *MHSRP Load Sharing*

Router A Configuration

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 10.0.0.1 255.255.255.0
Switch(config-if)# standby ip 10.0.0.3
Switch(config-if)# standby 1 priority 110
Switch(config-if)#  standby 1 preempt
Switch(config-if)# standby 2 ip 10.0.0.4
Switch(config-if)#  standby 2 preempt
Switch(config-if)# end
```

Router B Configuration

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 10.0.0.2 255.255.255.0
Switch(config-if)# standby ip 10.0.0.3
Switch(config-if)#  standby 1 preempt
Switch(config-if)# standby 2 ip 10.0.0.4
Switch(config-if)# standby 2 priority 110
Switch(config-if)#  standby 2 preempt
Switch(config-if)# end
```

Configuring HSRP Authentication and Timer: Example

This example shows how to configure word as the authentication string required to allow Hot Standby routers in group 1 to interoperate:

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/0/1
Switch(config-if)# no switchport
Switch(config-if)# standby 1 authentication word
Switch(config-if)# end
```

This example shows how to set the timers on standby group 1 with the time between hello packets at 5 seconds and the time after which a router is considered down to be 15 seconds:

```
Switch # configure terminal
Switch(config) # interface gigabitethernet1/0/1
Switch(config-if)# no switchport
Switch(config-if)# standby 1 ip
Switch(config-if)# standby 1 timers 5 15
Switch(config-if)# end
```

Configuring HSRP Groups and Clustering: Example

This example shows how to bind standby group my_hsrp to the cluster and enable the same HSRP group to be used for command switch redundancy and router redundancy. The command can only be executed on the cluster command switch. If the standby group name or number does not exist, or if the switch is a cluster member switch, an error message appears.

```
Switch # configure terminal
Switch(config) # cluster standby-group my_hsrp routing-redundancy
Switch(config-if)# end
```

Additional References for Configuring HSRP

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	*Command Reference (Catalyst 9600 Series Switches)*

Standards and RFCs

Standard/RFC	Title
RFC 2281	Cisco Hot Standby Router Protocol

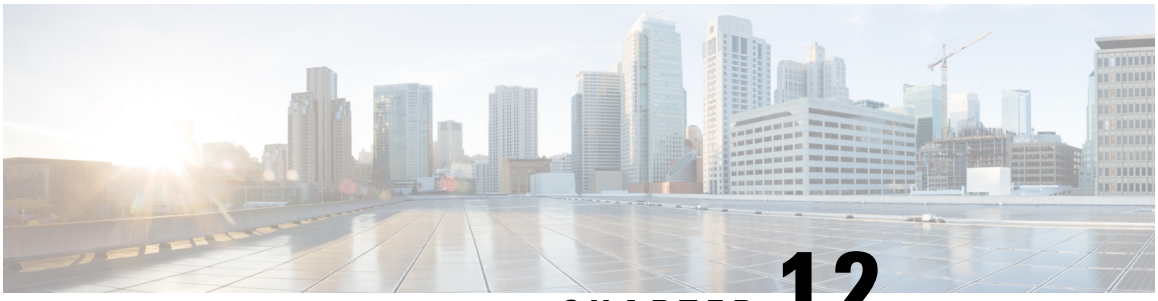
Feature History for HSRP

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 Gibraltar 16.11.1	HSRP	HSRP is Cisco's standard method of providing high network availability by providing first-hop redundancy for IP hosts on an IEEE 802 LAN configured with a default gateway IP address.
Cisco IOS XE Gibraltar 16.11.1	HSRP for IPv6	HSRP is an FHRP designed to allow for transparent failover of the first-hop IPv6 router.
Cisco IOS XE Cupertino 17.7.1	HSRP	This feature was implemented on the Cisco Catalyst 9600 Series Supervisor 2 Module.

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/



CHAPTER 12

Configuring NHRP

- Restrictions About Next Hop Resolution Protocol, on page 127
- Information About Next Hop Resolution Protocol, on page 127
- How to Configure Next Hop Resolution Protocol, on page 128
- Configuration Examples for Next Hop Resolution Protocol, on page 131
- Additional References for Configuring NHRP, on page 134
- Feature History for Next Hop Resolution Protocol , on page 134

Restrictions About Next Hop Resolution Protocol

Information About Next Hop Resolution Protocol

The following sections provide information about Next Hop Resolution Protocol (NHRP).

NHRP and NBMA Network Interaction

Most WAN networks are a collection of point-to-point links. Virtual tunnel networks (for example Generic Routing Encapsulation [GRE] tunnels) are also a collection of point-to-point links. To effectively scale the connectivity of these point-to-point links, they are usually grouped into a single or multilayer hub-and-spoke network. Multipoint interfaces (for example, GRE tunnel interfaces) can be used to reduce the configuration on a hub router in such a network. This resulting network is a NBMA network.

Because there are multiple tunnel endpoints that are reachable through a single multipoint interface, there needs to be a mapping from the logical tunnel endpoint IP address to the physical tunnel endpoint IP address, to forward packets out of the tunnel interfaces over this NBMA network. This mapping could be statically configured, but it is preferable if the mapping can be discovered or learned dynamically.

NHRP is an ARP-like protocol that alleviates these NBMA network problems. With NHRP, systems attached to an NBMA network dynamically learn the NBMA address of other systems that are part of the network, allowing these systems to directly communicate without requiring traffic to use an intermediate hop.

Routers, access servers, and hosts can use NHRP to discover the addresses of other routers and hosts connected to an NBMA network. Partially-meshed NBMA networks typically have multiple logical networks behind the NBMA network. In such configurations, packets traversing the NBMA network might have to make several hops over the NBMA network before arriving at the exit router (the router nearest the destination network).

NHRP Registration helps support these NBMA networks:

- NHRP Registration—NHRP allows Next Hop Clients (NHCs) to dynamically register with Next Hop Servers (NHSs). This registration function allows the NHCs to join the NBMA network without configuration changes on the NHSs, especially in cases where the NHC has a dynamic physical IP address or is behind a Network Address Translation (NAT) router that dynamically changes the physical IP address. In these cases, it would be impossible to preconfigure the logical (VPN IP address) to physical (NBMA IP) mapping for the NHC on the NHS.

Dynamically Built Hub-and-Spoke Networks

With NHRP, the NBMA network is initially laid out as a hub-and-spoke network that can have multiple hierarchical layers of NHCs as spokes and NHSs as hubs. The NHCs are configured with static mapping information to reach their NHSs and will connect to their NHS and send an NHRP registration to the NHS. This configuration allows the NHS to dynamically learn the mapping information for the spoke, reducing the configuration needed on the hub and allowing the spoke to obtain a dynamic NBMA (physical) IP address.

How to Configure Next Hop Resolution Protocol

The following sections provide configuration information about NHRP.

Enabling NHRP on an Interface

Perform this task to enable NHRP for an interface on a switch. In general, all NHRP stations within a logical NBMA network should be configured with the same network identifier.

The NHRP network ID is used to define the NHRP domain for an NHRP interface and differentiate between multiple NHRP domains or networks, when two or more NHRP domains (GRE tunnel interfaces) are available on the same NHRP node (switch). The NHRP network ID helps keep two NHRP networks (clouds) separate when both are configured on the same switch.

The NHRP network ID is a local-only parameter. It is significant only to the local switch and is not transmitted in NHRP packets to other NHRP nodes. For this reason the actual value of the NHRP network ID configured on a switch need not match the same NHRP network ID on another switch where both of these switches are in the same NHRP domain. As NHRP packets arrive on a GRE interface, they are assigned to the local NHRP domain in the NHRP network ID that is configured on that interface.

We recommend that the same NHRP network ID be used on the GRE interfaces on all switches that are in the same NHRP network. It is then easier to track which GRE interfaces are members of which NHRP network.

NHRP domains (network IDs) can be unique on each GRE tunnel interface on a switch. NHRP domains can span across GRE tunnel interfaces on a route. In this case the effect of using the same NHRP network ID on the GRE tunnel interfaces is to merge the two GRE interfaces into a single NHRP network.

Procedure

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

	Command or Action	Purpose
	`Switch> enable`	
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Switch(config)# interface tunnel 100`	Configures an interface and enters interface configuration mode.
Step 4	**ip address** *ip-address network-mask* **Example:** `Switch(config-if)# ip address 10.0.0.1 255.255.255.0`	Enables IP and gives the interface an IP address.
Step 5	**ip nhrp network-id** *number* **Example:** `Switch(config-if)# ip nhrp network-id 1`	Enables NHRP on the interface.
Step 6	**end** **Example:** `Switch(config)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring a GRE Tunnel for Multipoint Operation

Perform this task to configure a GRE tunnel for multipoint (NMBA) operation.

A tunnel network of multipoint tunnel interfaces can be considered of as an NBMA network. When multiple GRE tunnels are configured on the same switch, they must either have unique tunnel ID keys or unique tunnel source addresses.

Procedure

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

	Command or Action	Purpose
	`Switch# configure terminal`	
Step 3	**interface** *type number* **Example:** `Switch(config)# interface tunnel 100`	Configures an interface and enters interface configuration mode.
Step 4	**ip address** *ip-address* **Example:** `Switch(config-if)# ip address 172.16.1.1 255.255.255.0`	Configures an IP address for the interface.
Step 5	**ip mtu** *bytes* **Example:** `Switch(config-if)# ip mtu 1400`	Sets the maximum transmission unit (MTU) size of IP packets sent on an interface.
Step 6	**ip pim sparse-dense-mode** **Example:** `Switch(config-if)# ip pim sparse-dense-mode`	Enables Protocol Independent Multicast (PIM) on an interface and treats the interface in either sparse mode or dense mode of operation, depending on which mode the multicast group operates in.
Step 7	**ip nhrp map** *ip-address nbma-address* **Example:** `Switch(config-if)# ip nhrp map 172.16.1.2 10.10.10.2`	Statically configures the IP-to-nonbroadcast multiaccess (NBMA) address mapping of IP destinations connected to an NBMA network. • *ip-address*—IP address of the destinations reachable through the NBMA network. This address is mapped to the NBMA address. • *nbma-address*—NBMA address that is directly reachable through the NBMA network. The address format varies depending on the medium used. For example, ATM has a Network Service Access Point (NSAP) address, Ethernet has a MAC address, and Switched Multimegabit Data Service (SMDS) has an E.164 address. This address is mapped to the IP address.
Step 8	**ip nhrp map multicast** *nbma-address* **Example:** `Switch(config-if)# ip nhrp map multicast 10.10.10.2`	Configures nonbroadcast multiaccess (NBMA) addresses used as destinations for broadcast or multicast packets to be sent over a tunnel network.

	Command or Action	Purpose
Step 9	**ip nhrp network-id** *number* **Example:** `Switch(config-if)# ip nhrp network-id 1`	Enable the Next Hop Resolution Protocol (NHRP) on an interface. • *number*—Globally unique, 32-bit network ID from a nonbroadcast multiaccess (NBMA) network. The range is from 1 to 4294967295.
Step 10	**ip nhrp nhs** *nhs-address* **Example:** `Switch(config-if)# ip nhrp nhs 172.16.1.2`	Specifies the address of one or more NHRP servers. • *nhs-address*—Address of the next-hop server being specified.
Step 11	**tunnel source vlan** *interface-number* **Example:** `Switch(config-if)# tunnel source vlan 1`	Sets the source address for a tunnel interface
Step 12	**tunnel destination** *ip-address* **Example:** `Switch(config-if)# tunnel destination 10.10.10.2`	Sets the destination address for a tunnel interface.
Step 13	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

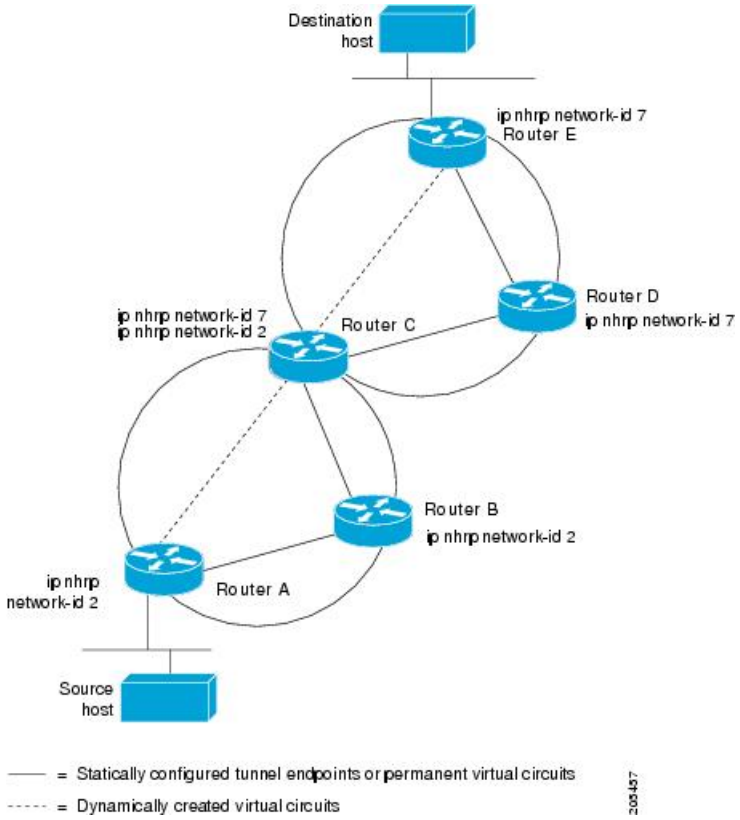
Configuration Examples for Next Hop Resolution Protocol

The following sections provide various configuration examples for NHRP.

Physical Network Designs for Logical NBMA Examples

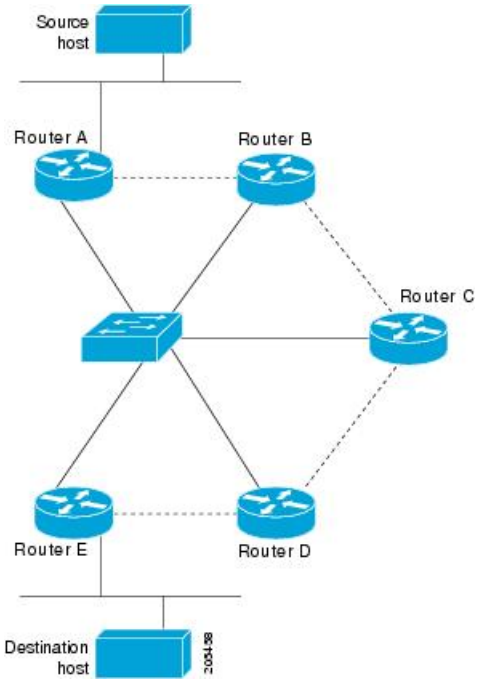
A logical NBMA network is considered the group of interfaces and hosts participating in NHRP and having the same network identifier. The figure below illustrates two logical NBMA networks (shown as circles) configured over a single physical NBMA network. Router A can communicate with routers B and C because they share the same network identifier (2). Router C can also communicate with routers D and E because they share network identifier 7. After address resolution is complete, router A can send IP packets to router C in one hop, and router C can send them to router E in one hop, as shown by the dotted lines.

Figure 13: Two Logical NBMA Networks over One Physical NBMA Network



The physical configuration of the five routers in the figure above might actually be that shown in the figure below. The source host is connected to router A and the destination host is connected to router E. The same switch serves all five routers, making one physical NBMA network.

Figure 14: Physical Configuration of a Sample NBMA Network



Refer again to the first figure above. Initially, before NHRP has resolved any NBMA addresses, IP packets from the source host to the destination host travel through all five routers connected to the switch before reaching the destination. When router A first forwards the IP packet toward the destination host, router A also generates an NHRP request for the IP address of the destination host. The request is forwarded to router C, whereupon a reply is generated. Router C replies because it is the egress router between the two logical NBMA networks.

Similarly, router C generates an NHRP request of its own, to which router E replies. In this example, subsequent IP traffic between the source and the destination still requires two hops to traverse the NBMA network, because the IP traffic must be forwarded between the two logical NBMA networks. Only one hop would be required if the NBMA network were not logically divided.

Example: GRE Tunnel for Multipoint Operation

With multipoint tunnels, a single tunnel interface may be connected to multiple neighboring switches. Unlike point-to-point tunnels, a tunnel destination need not be configured. In fact, if configured, the tunnel destination must correspond to an IP multicast address.

In the following example, switches A and B share an Ethernet segment. Minimal connectivity over the multipoint tunnel network is configured, thus creating a network that can be treated as a partially meshed NBMA network. Due to the static NHRP map entries, switch A knows how to reach switch B and vice versa.

The following example shows how to configure a GRE multipoint tunnel:

Switch A Configuration

```
Switch(config)# interface tunnel 100 !Tunnel interface configured for PIM traffic
Switch(config-if)# no ip redirects
Switch(config-if)# ip address 192.168.24.1 255.255.255.252
```

```
Switch(config-if)# ip mtu 1400
Switch(config-if)# ip pim sparse-dense-mode
Switch(config-if)# ip nhrp map 192.168.24.3 172.16.0.1 !NHRP may optionally be configured
to dynamically discover tunnel end points.
Switch(config-if)# ip nhrp map multicast 172.16.0.1
Switch(config-if)# ip nhrp network-id 1
Switch(config-if)# ip nhrp nhs 192.168.24.3
Switch(config-if)# tunnel source vlan 1
Switch(config-if)# tunnel destination 172.16.0.1
Switch(config-if)# end
```

Switch B Configuration

```
Switch(config)# interface tunnel 100
Switch(config-if)# no ip redirects
Switch(config-if)# ip address 192.168.24.2 255.255.255.252
Switch(config-if)# ip mtu 1400
Switch(config-if)# ip pim sparse-dense-mode
Switch(config-if)# ip nhrp map 192.168.24.4 10.10.0.3
Switch(config-if)# ip nhrp map multicast 10.10.10.3
Switch(config-if)# ip nhrp network-id 1
Switch(config-if)# ip nhrp nhs 192.168.24.4
Switch(config-if)# tunnel source vlan 1
Switch(config-if)# tunnel destination 10.10.10.3
Switch(config-if)# end
```

Additional References for Configuring NHRP

RFCs

RFC	Title
RFC 2332	*NBMA Next Hop Resolution Protocol (NHRP)*

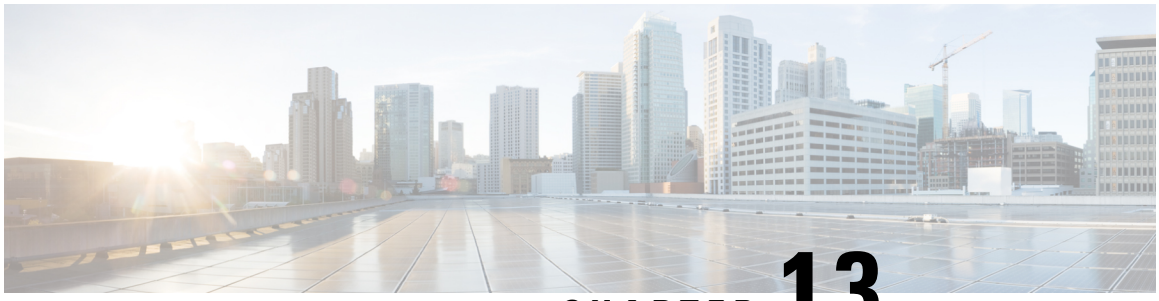
Feature History for Next Hop Resolution Protocol

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 Gibraltar 16.11.1	Next Hop Resolution Protocol	NHRP is an ARP-like protocol that dynamically maps a NBMA network instead of manually configuring all the tunnel end points. With NHRP, systems attached to an NBMA network can dynamically learn the NBMA (physical) address of the other systems that are part of that network, allowing these systems to directly communicate.

Use the Cisco Feature Navigator to find information about platform and software image support.



CHAPTER 13

Configuring Network Address Translation

- Restrictions For Network Address Translation, on page 137
- Information About Network Address Translation, on page 137
- Configuring Network Address Translation, on page 149
- Configuration Examples for Network Address Translation, on page 167
- Troubleshooting NAT, on page 168
- Feature History for Network Address Translation, on page 168

Restrictions For Network Address Translation

- When Flexible NetFlow and Network Address Translation (NAT) are configured on an interface:
  - Flexible NetFlow will display and export the actual flow details; but not the translated flow details. Application-level gateway (ALG) flow details are not part of the actual flow details that are exported.
  - If the ALG traffic gets translated through the CPU, Flexible NetFlow will display and export the translated flow details for the ALG traffic.

Information About Network Address Translation

The following sections provide information about Network Address Translation (NAT).

Network Address Translation

NAT is designed for IP address conservation. It enables private IP networks that use unregistered IP addresses to connect to the Internet. NAT operates on a device, usually connecting two networks together, and translates the private (not globally unique) addresses in the internal network into global routable addresses, before packets are forwarded onto another network.

NAT can be configured to advertise only one address for the entire network to the outside world. This ability provides additional security by effectively hiding the entire internal network behind that one address. NAT offers the dual functions of security and address conservation and is typically implemented in remote-access environments.

NAT is also used at the enterprise edge to allow internal users access to the Internet and to allow Internet access to internal devices such as mail servers.

Benefits of Configuring Network Address Translation

- Resolves the problem of IP depletion.

  NAT allows organizations to resolve the problem of IP address depletion when they have existing networks and need to access the Internet. Sites that do not yet possess Network Information Center (NIC)-registered IP addresses must acquire IP addresses, and if more than 254 clients are present or are planned, the scarcity of Class B addresses becomes a serious issue. NAT addresses these issues by mapping thousands of hidden internal addresses to a range of easy-to-get Class C addresses.

- Provides a layer of security by preventing the client IP address from being exposed to the outside network.

  Sites that already have registered IP addresses for clients on an internal network may want to hide those addresses from the Internet so that hackers cannot directly attack clients. With client addresses hidden, a degree of security is established. NAT gives LAN administrators complete freedom to expand Class A addressing, which is drawn from the reserve pool of the Internet Assigned Numbers Authority. The expansion of Class A addresses occurs within the organization without a concern for addressing changes at the LAN or the Internet interface.

- Cisco software can selectively or dynamically perform NAT. This flexibility allows network administrator to use RFC 1918 addresses or registered addresses.

- NAT is designed for use on a variety of devices for IP address simplification and conservation. In addition, NAT allows the selection of internal hosts that are available for translation.

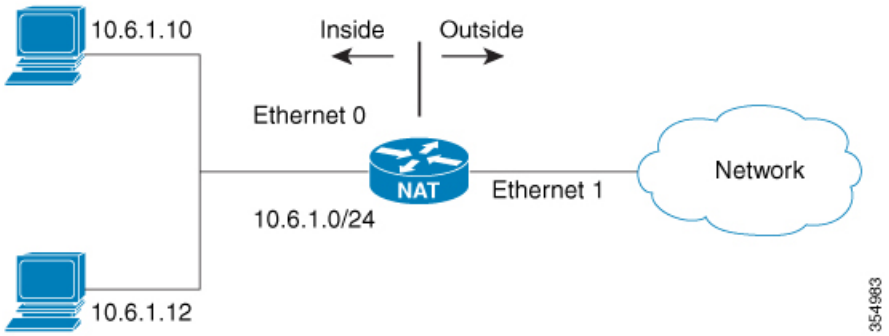
- A significant advantage of NAT is that it can be configured without requiring any changes to devices other than to those few devices on which NAT will be configured.

How Network Address Translation Works

A device that is configured with NAT will have at least one interface to the inside network and one to the outside network. In a typical environment, NAT is configured at the exit device between a stub domain and the backbone. When a packet leaves the domain, NAT translates the locally significant source address into a globally unique address. When a packet enters the domain, NAT translates the globally unique destination address into a local address. Multiple inside networks could be connected to the device and similarly there might exist multiple exit points from the device towards outside networks. If NAT cannot allocate an address because it has run out of addresses, it drops the packet and sends an Internet Control Message Protocol (ICMP) host unreachable packet to the destination.

Translation and forwarding are performed in the hardware switching plane, thereby improving the overall throughput performance. For more details on performance, refer the section on Performance and Scale Numbers for NAT.

Figure 15: NAT



Uses of Network Address Translation

NAT can be used for the following scenarios:

- To connect to the Internet when only a few of your hosts have globally unique IP address.

  NAT is configured on a device at the border of a stub domain (referred to as the inside network) and a public network such as the Internet (referred to as the outside network). NAT translates internal local addresses to globally unique IP addresses before sending packets to the outside network. As a solution to the connectivity problem, NAT is practical only when relatively few hosts in a stub domain communicate outside of the domain at the same time. When this is the case, only a small subset of the IP addresses in the domain must be translated into globally unique IP addresses when outside communication is necessary, and these addresses can be reused

- Renumbering:

  Instead of changing the internal addresses, which can be a considerable amount of work, you can translate them by using NAT.

Network Address Translation Inside and Outside Addresses

The term *inside* in a NAT context refers to networks owned by an organization that must be translated. When NAT is configured, hosts within this network will have addresses in one space (known as the local address space) that will appear to those outside the network as being in another space (known as the global address space).

Similarly, the term outside refers to those networks to which the stub network connects, and which are generally not under the control of an organization. Hosts in outside networks can also be subject to translation, and can thus have local and global addresses.

NAT uses the following definitions:

- Inside local address—an IP address that is assigned to a host on the inside network. The address is probably not a routable IP address assigned by NIC or service provider.
- Inside global address—a global routable IP address (assigned by the NIC or service provider) that represents one or more inside local IP addresses to the outside world.
- Outside local address—the IP address of an outside host as it appears to the inside network. Not necessarily a routable IP address, it is allocated from the address space that is routable on the inside.

- Outside global address—the IP address assigned to a host on the outside network by the owner of the host. The address is allocated from a globally routable address or network space.
- Inside Source Address Translation—translates an inside local address to inside global address.
- Outside Source Address Translation—translates the outside global address to outside local address.
- Static Port Translation—translates the IP address and port number of an inside/outside local address to the IP address and port number of the corresponding inside/outside global address.
- Static Translation of a given subnet—translates a specified range of subnets of an inside/outside local address to the corresponding inside/outside global address.
- Half Entry—represents a mapping between the local and global address/ports and is maintained in the translation database of NAT module. A half entry may be created statically or dynamically based on the configured NAT rule.
- Full Entry/Flow entry—represents a unique flow corresponding to a given session. In addition to the local to global mapping, it also maintains the destination information which fully qualifies the given flow. A Full entry is always created dynamically and maintained in the translation database of NAT module.

Note You can enable NAT on Layer 3 Multi-chassis EtherChannel (MEC) using the **interface port-channel** command.

VRF-Aware Network Address Translation

NAT is typically configured to operate across the default or global routing domain. As per this feature, the inside and outside NAT domains are associated with the default VRF space and the translations are effected accordingly. However, there are certain scenarios where NAT is required to operate in a VRF setting. One common scenario involves enabling shared service access for private networks that have overlapping address space. In such cases, the given private networks can be placed in different VRFs and global service access can be achieved by configuring VRF-aware NAT rules that map overlapping private address to unique global address. VRF-awareness enables NAT to carry out address and port translation by taking the VRF of the private networks into consideration.

VRF-aware NAT supports only VRF to Global translation of IP addresses. VRF to Global translation is between a NAT inside interface that is associated with a specific VRF and a NAT outside interface that is associated with the global VRF. Intra-VRF NAT translation (which involves the NAT-inside and NAT-outside interfaces of the same specific VRF) and Inter-VRF NAT translation (which involves NAT-inside and NAT-outside interfaces that are associated with different VRFs) are not supported. NAT behavior is undefined in such unsupported scenarios. We recommend that you deploy only the VRF to Global NAT translation in your network.

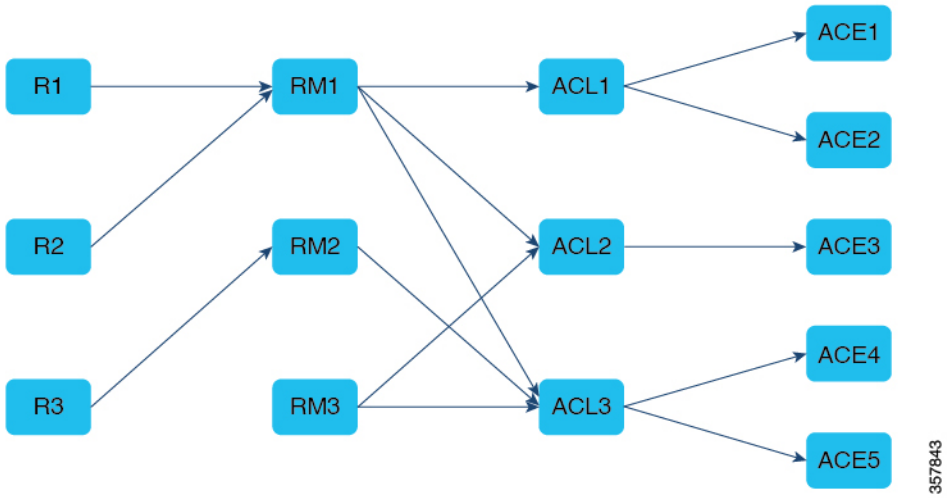
Route Map-based Network Address Translation

Route maps are similar to access control lists (ACLs) and allow users to define a set of match-criteria with associated actions. Route maps are more versatile and have additional capabilities that can address certain use cases that cannot be handled by ACLs. Route maps are widely used by various applications such as policy-based routing (PBR) and route-redistribution etc. For more information about route maps, see the

"Route Maps to Redistribute Routing Information" section in the *Cisco IOS XE IP Routing Configuration Guide*.

In the context of NAT, route maps are used to carry out destination-based translations, where the same local addresses are translated to different global addresses based on the flow destination. Route map-based static and dynamic translations are supported, and the same route map can be associated with multiple rules as illustrated in the figure below.

Figure 16: NAT Route Map



Route maps may include multiple permit-deny clauses and the corresponding address match criteria is specified through ACLs. Route map evaluation will yield a permit or deny disposition based on which the translation decision is made.

Unlike ACL-based NAT, route map-based NAT results in the creation of full flow-entries even when Address Only Translation (AOT) is enabled. This leads to relatively higher TCAM resource usage due to the programming of per-flow forward and reverse entries. To prevent this, enable the **nat scale** command which optimizes TCAM usage and improves the overall hardware scale.

Limitations of Route Map based Network Address Translation

- Route map support is limited to inside translations only. You can only configure static NAT/PAT and dynamic NAT/PAT. Outside translations, static network translations and interface-based static translations are not supported.
- Route map-based NAT supports only address-based match criteria. Next-Hop and interface-based match criteria are not supported.
- The TCAM usage increases when there are overlapping ACEs in route maps.
- In the case of static NAT, packets matching a deny clause get software switched if there is a subsequent permit clause within the same route map that allows the given packet.
- In the case of dynamic NAT, packets matching a deny ACL associated with a permit clause are hardware forwarded untranslated, even if there happens to be a subsequent permit clause that allows the given packet.

- In the case of dynamic non-overload NAT, packets originating from the outside domain destined to inside global addresses are translated.

These limitations apply only to the Cisco Catalyst 9600 Series Switches

- When you enable the **nat scale** command, the maximum number of route maps supported is four.
- If you enable the **nat scale** command after configuring the translation rules, there will be no change in the scale numbers. Configure the **nat scale** command before setting the route map rules for effective TCAM optimization.
- When the **nat scale** command is enabled, the show **ip nat translations** and the **show ip nat statistics** commands do not show all the NAT flows being translated and are not reliable. The **show platform software fed switch active nat flows-detail** command may give better visibility.

Types of Network Address Translation

You can configure NAT such that it will advertise only a single address for your entire network to the outside world. Doing this effectively hides the internal network from the world, giving you some additional security.

The types of NAT include:

- Static address translation (static NAT)—Allows one-to-one mapping between local and global addresses.
- Dynamic address translation (dynamic NAT)—Maps unregistered IP addresses to registered IP addresses from a pool of registered IP addresses.
- Overloading / PAT—Maps multiple unregistered IP addresses to a single registered IP address (many to one) using different Layer 4 ports. This method is also known as Port Address Translation (PAT). By using overloading, thousands of users can be connected to the Internet by using only one real global IP address.

Using Network Address Translation to Route Packets to the Outside Network (Inside Source Address Translation)

You can translate unregistered IP addresses into globally unique IP addresses when communicating outside your network.

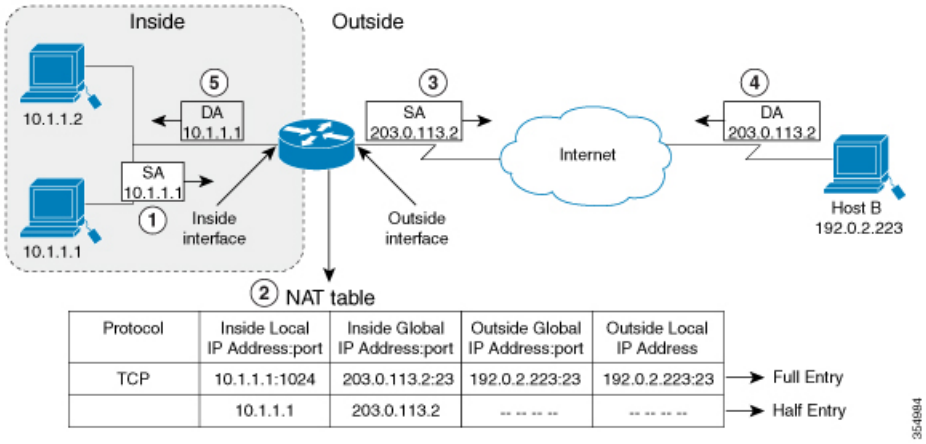
You can configure static or dynamic inside source address translation as follows:

- Static translation establishes a one-to-one mapping between the inside local address and an inside global address. Static translation is useful when a host on the inside must be accessible by a fixed address from the outside. Static translation can be enabled by configuring a static NAT rule as explained in the #unique_225 section.
- Dynamic translation establishes a mapping between an inside local address and a pool of global addresses dynamically. Dynamic translation can be enabled by configuring a dynamic NAT rule and the mapping is established based on the result of the evaluation of the configured rule at run-time. You can employ an Access Control List (ACL), both Standarad and Extended ACLs, to specify the inside local address. The inside global address can be specified through an address pool or an interface. Dynamic translation is enabled by configuring a dynamic rule as explained in the #unique_226 section.

You can configure both static and dynamic NAT rules on the device. It is possible for these rules to overlap with each other such that a given address is considered eligible for translation by both the rules. In such cases, a Static rule takes precedence over the Dynamic rule regardless of the order in which they have been configured.

The following figure illustrates a device that is translating a source address inside a network to a source address outside the network.

Figure 17: NAT Inside Source Translation



The following process describes the inside source address translation, as shown in the figure above:

1. The user at host 10.1.1.1 opens a connection to Host B in the outside network.
2. NAT module intercepts the corresponding packet and attempts to translate the packet.

   The following scenarios are possible based on the presence or absence of a matching NAT rule:

   - If a matching static translation rule exists, the packet gets translated to the corresponding inside global address. Otherwise, the packet is matched against the dynamic translation rule and in the event of a successful match, it gets translated to the corresponding inside global address. The NAT module inserts a fully qualified flow entry corresponding to the translated packet, into its translation database. This facilitates fast translation and forwarding of the packets corresponding to this flow, in either direction.
   - The packet gets forwarded without any address translation in the absence of a successful rule match.
   - The packet gets dropped in the event of failure to obtain a valid inside global address even-though we have a successful rule match.

Note If an ACL is employed for dynamic translation, NAT evaluates the ACL and ensures that only the packets that are permitted by the given ACL are considered for translation.

3. The device replaces the inside local source address of host 10.1.1.1 with the inside global address of the translation, 203.0.113.2, (only the packet-relevant checksums get updated and all other fields in the packet remain unchanged) and forwards the packet.

4. The NAT module inserts a fully qualified flow entry corresponding to the translated packet flow, into its translation database. This facilitates fast translation and forwarding of packets corresponding to the flow in either direction.
5. Host B receives the packet and responds to host 10.1.1.1 by using the inside global IP destination address (DA) 203.0.113.2
6. The response packet from host B would be destined to the inside global address and the NAT module intercepts this packet and translates it back to the corresponding inside local address with the help of the flow entry that has been setup in the translation database.

Host 10.1.1.1 receives the packet and continues the conversation. The device performs Steps 2 to 5 for each packet that it receives.

Outside Source Address Translation

You can translate the source address of the IP packets that travel from outside of the network to inside the network. This type of translation is usually employed in conjunction with inside source address translation to interconnect overlapping networks.

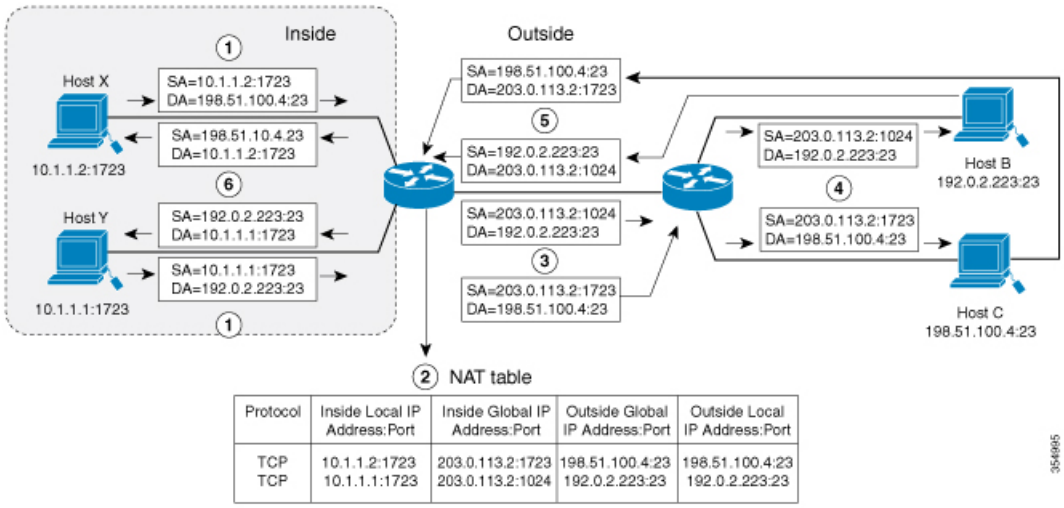
This process is explained in the section on #unique_228

Port Address Translation (PAT)

You can conserve addresses in the inside global address pool by allowing a device to use one global address for many local addresses and this type of NAT configuration is called overloading or port address translation. When overloading is configured, the device maintains enough information from higher-level protocols (for example, TCP or UDP port numbers) to translate the global address back to the correct local address. When multiple local addresses map to one global address, the TCP or UDP port numbers of each inside host distinguish between the local addresses.

The figure below illustrates a NAT operation when an inside global address represents multiple inside local addresses. The TCP port numbers act as differentiators.

Figure 18: PAT / NAT Overloading Inside Global Addresses



The device performs the following process in the overloading of inside global addresses, as shown in the figure above. Both Host B and Host C believe that they are communicating with a single host at address 203.0.113.2. Whereas, they are actually communicating with different hosts; the port number is the differentiator. In fact, many inside hosts can share the inside global IP address by using many port numbers.

1. The user at Host Y opens a connection to Host B and the user at Host X opens a connection to Host C.
2. NAT module intercepts the corresponding packets and attempts to translate the packets.

   Based on the presence or absence of a matching NAT rule the following scenarios are possible:

   - If a matching static translation rule exists, then it takes precedence and the packets are translated to the corresponding global address. Otherwise, the packets are matched against dynamic translation rule and in the event of a successful match, they are translated to the corresponding global address. NAT module inserts a fully qualified flow entry corresponding to the translated packets, into its translation database, to facilitate fast translation and forwarding of the packets corresponding to this flow, in either direction.
   - The packets get forwarded without any address translation in the absence of a successful rule match.
   - The packets get dropped in the event of failure to obtain a valid inside global address even though we have a successful rule match.
   - As this is a PAT configuration, transport ports help translate multiple flows to a single global address. (In addition to source address, the source port is also subjected to translation and the associated flow entry maintains the corresponding translation mappings.)

3. The device replaces inside local source address/port 10.1.1.1/1723 and 10.1.1.2/1723 with the corresponding selected global address/port 203.0.113.2/1024 and 203.0.113.2/1723 respectively and forwards the packets.
4. Host B receives the packet and responds to Host Y by using the inside global IP address 203.0.113.2, on port 1024. Host C receives the packet and responds to Host X using the inside global IP adress 203.0.113.2, on port 1723.
5. When the device receives the packets with the inside global IP address, it performs a NAT table lookup; the inside global address and port, and the outside address and port as keys; translates the addresses to the inside local addresses 10.1.1.1:1723 / 10.1.1.2:1723 and forwards the packets to Host Y and Host X respectively.

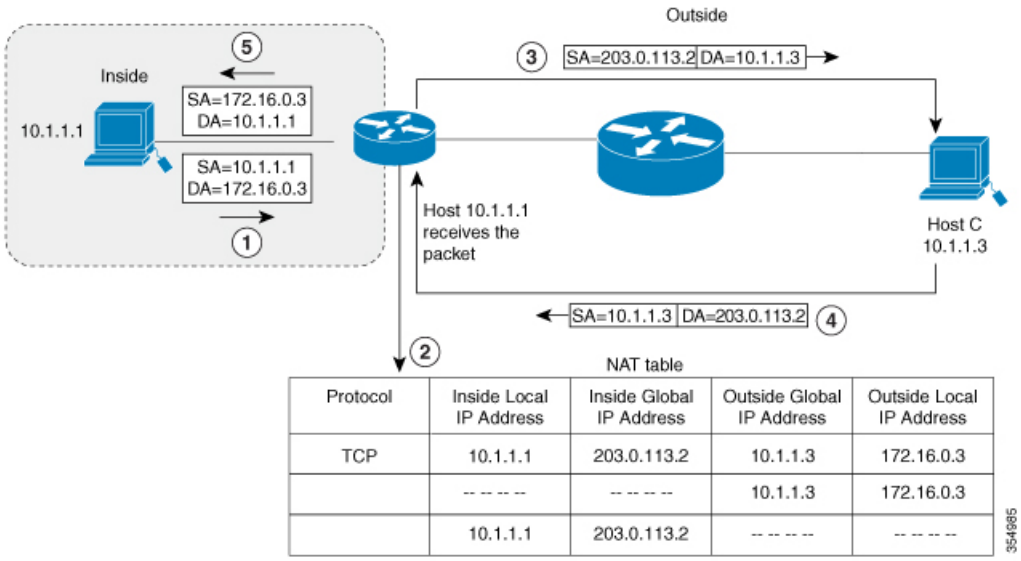
Host Y and Host X receive the packet and continue the conversation. The device performs Steps 2 to 5 for each packet it receives.

Overlapping Networks

Use NAT to translate IP addresses if the IP addresses that you use are neither legal nor officially assigned. Overlapping networks result when you assign an IP address to a device on your network that is already legally owned and assigned to a different device on the Internet or outside the network.

The following figure depicts overlapping networks: the inside network and outside network both have the same local IP addresses (10.1.1.x). You need network connectivity between such overlapping address spaces with one NAT device to translate the address of a remote peer (10.1.1.3) to a different address from the perspective of the inside.

Figure 19: NAT Translating Overlapping Addresses



Protocol	Inside Local IP Address	Inside Global IP Address	Outside Global IP Address	Outside Local IP Address
TCP	10.1.1.1	203.0.113.2	10.1.1.3	172.16.0.3
	-- -- -- --	-- -- -- --	10.1.1.3	172.16.0.3
	10.1.1.1	203.0.113.2	-- -- -- --	-- -- -- --

Notice that the inside local address (10.1.1.1) and the outside global address (10.1.1.3) are in the same subnet. To translate the overlapping address, first, the inside source address translation happens with the inside local address getting translated to 203.0.113.2 and a half entry is created in the NAT table. On the Receiving side, the outside source address is translated to 172.16.0.3 and another half entry is created. The NAT table is then updated with a full entry of the complete translation.

The following steps describe how a device translates overlapping addresses:

1. Host 10.1.1.1 opens a connection to 172.16.0.3.
2. The NAT module sets up the translation mapping of the inside local and global addresses to each other and the outside global and local addresses to each other
3. The Source Address (SA) is replaced with inside global address and the Destination Address (DA) is replaced with outside global address.
4. Host C receives the packet and continues the conversation.
5. The device does a NAT table lookup, replaces the DA with inside local address, and replaces the SA with outside local address.
6. Host 10.1.1.1 receives the packet and the conversation continues using this translation process.

Address Only Translation

Address only Translation (AOT) functionality can be employed in situations that require only the address fields to be translated and not the transport ports. In such settings, enabling AOT functionality significantly increases the number of flows that can be translated and forwarded in the hardware at line-rate. This improvement is brought about by optimizing the usage of various hardware resources associated with translation and forwarding. A typical NAT focused resource allocation scheme sets aside 15500 TCAM entries for performing hardware translation. This places a strict upper limit on the number of flows that can be translated and forwarded at line-rate. Under AOT scheme, the usage of TCAM resource is highly optimized thereby

enabling the accommodation of more number of flows in the TCAM tables and this provides a significant improvement in the hardware translation and forwarding scale. AOT can be very effective in situations where majority of the flows are destined to a single or a small set of destinations. Under such favourable conditions, AOT can potentially enable line-rate translation and forwarding of all the flows originating from the given end-point(s). AOT functionality is disabled by default. It can be enabled using the **no ip nat create flow-entries** command. The existing dynamic flow can be cleared using the **clear ip nat translation** command. The AOT feature can be disabled using the **ip nat create flow-entries** command.

Restrictions for Address Only Translation

- AOT feature is expected to function correctly only in translation scenarios corresponding to simple inside static and inside dynamic rules. The simple static rule must be of the type **ip nat inside source static** *local-ip global-ip*, and the dynamic rule must be of the type **ip nat inside source list** *access-list* **pool** *name*.
- When AOT is enabled, the **show ip nat translations** command will not give visibility into all the NAT flows being translated and forwarded.

Limitations of Network Address Translation

- There are certain NAT operations that are currently not supported in the hardware data plane. The following are such operations that are carried out in the relatively slower Software data plane:
  - Translation of Internet Control Message Protocol (ICMP) packets.
  - Translation of packets that require application layer gateway (ALG) processing.
  - Packets that require both inside and outside translation.
- The maximum number of sessions that can be translated and forwarded in the hardware in an ideal setting is limited to 7750. Additional flows that require translation are handled in the software data plane at a reduced throughput.

Note Each translation consumes two entries in TCAM.

- For NAT traffic, the CPU Queue bandwidth limitation is 2000 packets. Packets that exceed this limit will be dropped.
- A configured NAT rule might fail to get programmed into the hardware owing to resource constraint. This could result in packets that correspond to the given rule to get forwarded without translation.
- ALG support is currently limited to FTP, TFTP, and ICMP protocols. Also, although TCP SYN, TCP FIN, and TCP RST are not part of ALG traffic, they are processed as part of ALG traffic.
- Dynamically created NAT flows age out after a period of inactivity.
- Policy Based Routing (PBR) and NAT are not supported on the same interface. PBR and NAT work together only if they are configured on different interfaces.
- NAT does not support translation of fragmented packets.

- Bidirectional Forwarding Detection (BFD) sessions may fail if they are configured to operate using the same address that is used for dynamic NAT. To avoid a conflict that arises when both BFD and Dynamic NAT are configured on the device, use an address that does not overlap with NAT. If you must configure BFD and dynamic NAT overloading on the same interface, deploy a pool-based dynamic NAT overload configuration. Ensure that you do not use the chosen NAT pool address for BFD even in this scenario.
- Equal-cost multi-path routing (ECMP) is not supported with NAT.
- When Flexible NetFlow and Network Address Translation (NAT) are configured on an interface:
  - Flexible NetFlow will display and export the actual flow details; but not the translated flow details. Application-level gateway (ALG) flow details are not part of the actual flow details that are exported.
  - If the ALG traffic gets translated through the CPU, Flexible NetFlow will display and export the translated flow details for the ALG traffic.
- Explicit deny access control entry (ACE) in NAT ACL is not supported. Only explicit permit ACE is supported.
- NETCONF configuration fails if it is configured to operate using the same IP address that is used for configuring NAT using interface overload.
- NAT is not supported over GRE tunnels.
- When both the ingress and egress NAT interfaces are on the same switch, hardware NAT entries in TCAM may go out of sync on the remote or standby switch after stateful switchover (SSO). When AOT is enabled, this can occur on the remote switch after NAT timeout.

Performance and Scale Numbers for Network Address Translation

NAT module is capable of performing translation and forwarding in the hardware at line-rate, by programming the relevant hardware tables with the forwarding and rewrite information. You can configure a NAT-focused resource allocation scheme to obtain increased NAT throughput.

Configure SDM template NAT to achieve better performance and scale number. Refer Configuring Switch Database Management (SDM) Template, on page 161

The maximum number of TCAM flows that are available in the hardware is 15500.

Note Starting from Cisco IOS XE Bengaluru 17.5.1, you can enhance the NAT scale support using the **natscale** command in global configuration mode. When enabled, the NAT throughput is below the line rate of NAT translated traffic. However, the maximum TCAM flow supported will remain at 15500.

Note Using Address Only Translation optimizes the handling of flows and enhances the scale of the NAT feature.

Using Application-Level Gateways with Network Address Translation

NAT performs translation services on any TCP/UDP traffic that does not carry source and destination IP addresses in the application data stream. Protocols that do not carry the source and destination IP addresses include HTTP, TFTP, telnet, archie, finger, Network Time Protocol (NTP), Network File System (NFS), remote login (rlogin), remote shell (rsh) protocol, and remote copy (rcp).

NAT Application-Level Gateway (ALG) enables certain applications that carry address/port information in their payloads to function correctly across NAT domains. In addition to the usual translation of address/ports in the packet headers, ALGs take care of translating the address/ports present in the payload and setting up temporary mappings.

Best Practices for NAT Configuration

- In cases where both static and dynamic rules are configured, ensure that the local addresses specified in the rules do not overlap. If such an overlap is possible, then the ACL associated with the dynamic rule should exclude the corresponding addresses used by the static rule. Similarly, there must not be any overlap between the global addresses as this could lead to undesired behavior.
- Do not employ loose filtering such as **permit ip any any** in an ACL associated with NAT rule as this could result in unwanted packets being translated.
- Do not share an address pool across multiple NAT rules.
- Do not define the same inside global address in Static NAT and Dynamic Pool. This action can lead to undesirable results.
- Exercise caution while modifying the default timeout values associated with NAT. Small timeout values could result in high CPU usage.
- Exercise caution while manually clearing the translation entries as this could result in the disruption of application sessions.
- Follow these steps before you make NAT configuration changes during active translations.
    - Stop the ingress and egress of traffic matching the given configuration. This may require applying an appropriate ACL filter or shutting down the given interfaces.
    - Clear any existing translation entries that correspond to the given configuration.
    - Make the desired configuration change and re-enable the stopped traffic.

Configuring Network Address Translation

The tasks described in this section will help you configure NAT. Based on the desired configuration, you may need to configure more than one task.

Configuring Static Translation of Inside Source Addresses

Configure static translation of inside source address to allow one-to-one mapping between an inside local address and an inside global address. Static translation is useful when a host on the inside must be accessible by a fixed address from the outside.

Procedure

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

Step 3	Use any of the following commands depending on the requirement:
	• **ip nat inside source static** *local-ip global-ip*
	`Switch(config)# ip nat inside source static 10.10.10.1 172.16.131.1`
	• **ip nat inside source static** *protocol local-ip port global-ip port*
	`Switch(config)# ip nat inside source static tcp 10.10.10.1 1234 172.16.131.1 5467`
	• **ip nat inside source static network** *local-ip global-ip* {**prefix_len** *len* | **subnet** *subnet-mask*}
	`Switch(config)# ip nat inside source static network 10.10.10.1 172.16.131.1 prefix_len 24`
	• **ip nat inside source static** *local-ip global-ip* **vrf** *vrf-name*
	`Switch(config)# ip nat inside source static 10.10.10.1 172.16.131.1 vrf vrf1`
		• Establishes static translation between an inside local address and an inside global address.
		• Establishes a static port translation between an inside local address and an inside global address.
		• Allows a static translation mapping for an entire subnet without the need for specifying multiple individual translation rules. You can specify the translation mapping for the desired subnet. The actual translation is performed by translating the network portion of the address with the host the portion remaining unchanged.
		• Makes the static translation VRF aware and associates the given rule with the specified VRF.

Step 4	**interface** *type number* **Example:** `Switch(config)# interface ethernet 1`	Specifies an interface and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* **[secondary]** **Example:**	Sets a primary IP address for an interface.

	Command or Action	Purpose
	`Switch(config-if)# ip address 10.114.11.39 255.255.255.0`	
Step 6	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 7	**exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	**interface** *type number* **Example:** `Switch(config)# interface gigabitethernet 0/0/0`	Specifies a different interface and enters interface configuration mode.
Step 9	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)# ip address 172.31.232.182 255.255.255.240`	Sets a primary IP address for an interface.
Step 10	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 11	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Dynamic Translation of Inside Source Addresses

Dynamic translation establishes a mapping between an inside local address and a pool of global addresses dynamically. Dynamic translation can be enabled by configuring a dynamic NAT rule and the mapping is established based on the result of the evaluation of the configured rule at run-time. You can employ an ACL to specify the inside local address and the inside global address can be specified through an address pool or an interface.

Dynamic translation is useful when multiple users on a private network need to access the Internet. The dynamically configured pool IP address may be used as needed and is released for use by other users when access to the internet is no longer required.

Procedure

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

	Command or Action	Purpose
	`Switch> enable`	
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* **netmask** *netmask* | **prefix-length** *prefix-length* **Example:** `Switch(config)# ip nat pool net-208 172.16.233.208 172.16.233.223 prefix-length 28`	Defines a pool of global addresses to be allocated as needed.
Step 4	**access-list** *access-list-number* **permit** *source [source-wildcard]* **Example:** `Switch(config)# access-list 1 permit 192.168.34.0 0.0.0.255`	Defines a standard access list permitting those addresses that are to be translated.
Step 5	**ip nat inside source list** *access-list-number* **pool** *name* **vrf** *vrf-name* **Example:** `Switch(config)# ip nat inside source list 1 pool net-208`	Establishes dynamic source translation, specifying the access list defined in Step 4. Using the **vrf** keyword makes the dynamic translation VRF aware and associates the given rule with the specified VRF.
Step 6	**interface** *type number* **Example:** `Switch(config)# interface ethernet 1`	Specifies an interface and enters interface configuration mode.
Step 7	**ip address** *ip-address mask* **Example:** `Switch(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for the interface.
Step 8	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 9	**exit** **Example:** `Switch(config-if)#exit`	Exits the interface configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Switch(config)# interface ethernet 0`	Specifies an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 11	**ip address** *ip-address mask* **Example:** `Switch(config-if)# ip address 172.16.232.182 255.255.255.240`	Sets a primary IP address for the interface.
Step 12	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 13	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Port Address Translation

Configuring Port Address Translation by Overloading of Global Addresses

NAT module supports dynamic PAT configurations through address pools and interface, as described in the following tasks.

Perform the following task to allow your internal users access to the Internet and conserve addresses in the inside global address pool using overloading of global addresses.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* **netmask** *netmask* | **prefix-length** *prefix-length* **Example:** `Switch(config)# ip nat pool net-208 192.168.202.129 192.168.202.158 netmask 255.255.255.224`	Defines a pool of global addresses to be allocated as needed.
Step 4	**access-list** *access-list-number* **permit** *source [source-wildcard]* **Example:**	Defines a standard access list permitting those addresses that are to be translated. The access list must permit only those addresses that are to be translated. (Remember

	Command or Action	Purpose
	`Switch(config)# access-list 1 permit 192.168.201.30 0.0.0.255`	that there is an implicit "deny all" at the end of each access list.) Use of an access list that is too permissive can lead to unpredictable results.
Step 5	**ip nat inside source list** *access-list-number* **pool** *name* [**vrf** *vrf-name*] **overload** **Example:** `Switch(config)# ip nat inside source list 1 pool net-208 overload`	Establishes dynamic source translation with overloading, specifying the access list defined in Step 4. Using the **vrf** keyword makes the dynamic translation VRF aware and associates the given rule with the specified VRF.
Step 6	**interface** *type number* **Example:** `Switch(config)# interface ethernet 1`	Specifies an interface and enters interface configuration mode.
Step 7	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)# ip address 192.168.201.1 255.255.255.240`	Sets a primary IP address for an interface.
Step 8	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 9	**exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Switch(config)# interface ethernet 0`	Specifies a different interface and enters interface configuration mode.
Step 11	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)#  ip address 192.168.201.29 255.255.255.240`	Sets a primary IP address for an interface.
Step 12	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 13	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Port Address Translation by Overloading an Interface

Perform the following task to allow your internal users access to the Internet and conserve addresses in the inside global address pool by overloading an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *access-list-number* **permit** *source [source-wildcard]* **Example:** `Switch(config)# access-list 1 permit 192.168.201.30 0.0.0.255`	Defines a standard access list permitting those addresses that are to be translated. The access list must permit only those addresses that are to be translated. (Note that there is an implicit "deny all" at the end of each access list.) Use of an access list that is too permissive can lead to unpredictable results.
Step 4	**ip nat inside source list** *access-list-number* **interface** *name* **overload** **Example:** `Switch(config)# ip nat inside source list 1 interface gigabitethernet0/0/2 overload`	Establishes dynamic source translation with overloading, specifying the access list defined in Step 3. **Note** For overloading, you can choose any Layer 3 interface that has a valid IP address and is in operational state.
Step 5	**interface** *type number* **Example:** `Switch(config)# interface gigabitethernet0/0/1`	Specifies an interface and enters interface configuration mode.
Step 6	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)# ip address 192.168.201.1 255.255.255.240`	Sets a primary IP address for an interface.
Step 7	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**interface** *type number* **Example:** `Switch(config)# interface gigabitethernet0/0/2`	Specifies a different interface and enters interface configuration mode.
Step 10	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)#  ip address 192.168.201.29 255.255.255.240`	Sets a primary IP address for an interface.
Step 11	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 12	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Network Address Translation of External IP Addresses Only

By default, NAT translates the addresses embedded in the packet pay-load as explained in Using Application-Level Gateways with Network Address Translation, on page 149 section. There might be situations where the translation of the embedded address is not desirable and in such cases, NAT can be configured to translate the external IP address only.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static network** *local-ip global-ip* [**no-payload**]}	Disables the network packet translation on the inside host device.

	Command or Action	Purpose
	Example: `Device(config)# ip nat inside source static network 10.1.1.1 192.168.251.0/24 no-payload`	
Step 4	**ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static** {**tcp** | **udp**} *local-ip local-port global-ip global-port* [**no-payload**]} **Example:** `Device(config)# ip nat inside source static tcp 10.1.1.1 2000 192.168.1.1 2000 no-payload`	Disables port packet translation on the inside host device.
Step 5	**ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static** [**network**] *local-network-mask global-network-mask* [**no-payload**]} **Example:** `Device(config)# ip nat inside source static 10.1.1.1 192.168.1.1 no-payload`	Disables packet translation on the inside host device.
Step 6	**ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** *local-ip global-ip* [**no-payload**]} **Example:** `Device(config)# ip nat outside source static 10.1.1.1 192.168.1.1 no-payload`	Disables packet translation on the outside host device.
Step 7	**ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** {**tcp** | **udp**} *local-ip local-port global-ip global-port* [**no-payload**]} **Example:** `Device(config)# ip nat outside source static tcp 10.1.1.1 20000 192.168.1.1 20000 no-payload`	Disables port packet translation on the outside host device.
Step 8	**ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** [**network**] *local-network-mask global-network-mask* [**no-payload**]} **Example:** `Device(config)# ip nat outside source static network 10.1.1.1 192.168.251.0/24 no-payload`	Disables network packet translation on the outside host device.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 10	**show ip nat translations** [**verbose**] **Example:** `Device# show ip nat translations`	Displays active NAT.

Configuring Translation of Overlapping Networks

Configure static translation of overlapping networks if your IP addresses in the stub network are legitimate IP addresses belonging to another network and you want to communicate with those hosts or routers using static translation.

Note For a successful NAT outside translation, the device should be configured with a route for the outside local address. You can configure the route either manually or using the **add-route** option associated with **ip nat outside source** {**static** | **list**} command. We recommend that you use the **add-route** option to enable automatic creation of the route.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	
Step 3	**ip nat inside source static** *local-ip global-ip* **Example:** `Switch(config)# ip nat inside source static 10.1.1.1 203.0.113.2`	Establishes static translation between an inside local address and an inside global address.
Step 4	**ip nat outside source static** *local-ip global-ip* **Example:** `Switch(config)# ip nat outside source static 172.16.0.3 10.1.1.3`	Establishes static translation between an outside local address and an outside global address.
Step 5	**interface** *type number* **Example:**	Specifies an interface and enters interface configuration mode.

	Command or Action	Purpose
	`Switch(config)# interface ethernet 1`	
Step 6	**ip address** *ip-address mask* **Example:** `Switch(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for an interface.
Step 7	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Marks the interface as connected to the inside.
Step 8	**exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**interface** *type number* **Example:** `Switch(config)# interface ethernet 0`	Specifies a different interface and enters interface configuration mode.
Step 10	**ip address** *ip-address mask* **Example:** `Switch(config-if)#  ip address 172.16.232.182 255.255.255.240`	Sets a primary IP address for an interface.
Step 11	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Marks the interface as connected to the outside.
Step 12	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Address Translation Timeouts

You can configure address translation timeouts based on your NAT configuration.

By default, dynamically created translation entries time-out after a period of inactivity to enable the efficient use of various resources. You can change the default values on timeouts, if necessary. The following are the default time-out configurations associated with major translation types :

- Established TCP sessions: 24 hours
- UDP flow: 5 minutes
- ICMP flow: 1 minute

The default timeout values are adequate to address the timeout requirements in most of the deployment scenarios. However, these values can be adjusted/fine-tuned as appropriate. It is recommended not to configure

very small timeout values (less than 60 seconds) as it could result in high CPU usage. Refer the Best Practices for NAT Configuration, on page 149 section for more information.

Based on your configuration, you can change the timeouts described in this section.

- If you need to quickly free your global IP address for a dynamic configuration, configure a shorter timeout than the default timeout, by using the **ip nat translation timeout** command. However, the configured timeout should be longer than the other timeouts configured using commands specified in the following steps.
- If a TCP session is not properly closed by a finish (FIN) packet from both sides or during a reset, change the default TCP timeout by using the **ip nat translation tcp-timeout** command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**ip nat translation** *seconds* **Example:** `Switch(config)# ip nat translation 300`	(Optional) Changes the amount of time after which NAT translations time out. The default timeout is 24 hours, and it applies to the aging time for half-entries.
Step 4	**ip nat translation udp-timeout** *seconds* **Example:** `Switch(config)# ip nat translation udp-timeout 300`	(Optional) Changes the UDP timeout value.
Step 5	**ip nat translation tcp-timeout** *seconds* **Example:** `Switch(config)# ip nat translation tcp-timeout 2500`	(Optional) Changes the TCP timeout value. The default is 24 hours.
Step 6	**ip nat translation finrst-timeout** *seconds* **Example:** `Switch(config)# ip nat translation finrst-timeout 45`	(Optional) Changes the finish and reset timeout value. finrst-timeout—The aging time after a TCP session receives both finish-in (FIN-IN) and finish-out (FIN-OUT) requests or after the reset of a TCP session.
Step 7	**ip nat translation icmp-timeout** *seconds* **Example:**	(Optional) Changes the ICMP timeout value.

	Command or Action	Purpose
	`Switch(config)# ip nat translation icmp-timeout 45`	
Step 8	**ip nat translation syn-timeout** *seconds* **Example:** `Switch(config)# ip nat translation syn-timeout 45`	(Optional) Changes the synchronous (SYN) timeout value. The synchronous timeout or the aging time is used only when a SYN request is received on a TCP session. When a synchronous acknowledgment (SYNACK) request is received, the timeout changes to TCP timeout.
Step 9	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Switch Database Management (SDM) Template

Use SDM templates to configure system resources to optimize support for NAT.

After you set 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.

Follow these steps to set the SDM template to maximize NAT usage:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 2	**sdm prefer nat** **Example:** `Switch(config)# sdm prefer nat`	Specifies the SDM template to be used on the switch. Starting from Cisco IOS XE Gibraltar 16.12.1, this template is available under the network-advantage license. On all earlier releases, it is available under the DNA Advantage license.
Step 3	**end** **Example:** `Switch(config)# end`	Returns to the privileged EXEC mode.

	Command or Action	Purpose
Step 4	**write memory** **Example:** `Switch# write memory`	Save the current configuration before reload.
Step 5	**reload** **Example:** `Switch# reload`	Reloads the operating system.

Configuring a Static Rule Using a Route Map

Procedure

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

Step 3	**ip access-list extended** {*acl-num* | *acl-name*}
	Example:
	`Switch(config)# ip access-list extended ACL1`
	`Switch(config-ext-nacl)#`
		Enables extended ACL configuration mode.
		• *acl-name*: Specifies the access list using an alphanumeric string to which all commands entered from ACL configuration mode is applicable.
		• *acl-name*: Specifies the access list using numeric indentifier to which all commands entered from ACL configuration mode is applicable. The range is from 100 to 199. The extended range is from 2000 to 2699.

Step 4	**access-list** *seq-num* **ip** *ip-addresses*
	Example:
	`Switch(config-ext-nacl)# access-list 10 permit ip 1.1.1.0 0.0.0.255 8.8.8.0 0.0.0.255`
	`Switch# access-list 20 permit ip 1.1.1.0 0.0.0.255 8.8.8.2 0.0.0.255`
		Permits IP traffic between the specified IP addresses.

Step 5	**exit** **Example:**	Exits ACL configuration mode.

	Command or Action	Purpose
	`Switch(config-ext-nacl)# exit` `Switch(config)#`	

Step 6	**route-map***name* {**permit** | **deny** | *seq-num* | *ordering-seq*}
	Example:
	`Switch(config)# route-map RM1 permit 10`
	`Switch(config-route-map)#`
		Associates ACL with a route map.
		• *name*: Specifies the route-map name.
		• *seq-num*: Specifies sequence number to insert or delete from existing route-map entry. The range is from 0 to 65535.
		• *ordering-seq*: Specifies ordered sequence number to insert or delete from existing route-map entry.
		• **permit**: Permits set operations of the route-map.
		• **deny**: Denies set operations of the route-map.

Step 7	**match ip-address** *acl-name* **Example:** `Switch(config-route-map)# match ip address ACL1`	Configures the specified ACL's policies to the route map.
Step 8	**ip nat inside source static** *local-ip global-ip* **route-map** *name* **Example:** `Switch(config-route-map)# ip nat inside source static 1.1.1.10 20.20.20.20 route-map RM1`	Associates the route map with a static rule.

Configuring a Dynamic Rule Using a Route Map

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**ip access-list extended** {*acl-num* | *acl-name*}	Enables extended ACL configuration mode.

	Command or Action	Purpose

	Example:
	`Switch(config)# ip access-list extended ACL2`
	`Switch(config-ext-nacl)#`
		• *acl-name*: Specifies the access list using an alphanumeric string to which all commands entered from ACL configuration modeis applicable.
		• *acl-name*: Specifies the access list using numeric indentifier to which all commands entered from ACL configuration mode is applicable. The range is from 100 to 199. The extended range is from 2000 to 2699.

Step 4	**access-list** *seq-num* **ip** *ip-addresses* **Example:** `Switch(config-ext-nacl)# access-list 20 permit ip 1.1.1.0 0.0.0.255 8.8.8.0 0.0.0.255`	Permits IP traffic between the specified IP addresses.

Step 5	**exit**
	Example:
	`Switch(config-ext-nacl)# exit`
	`Switch(config)#`
		Exits ACL configuration mode.

Step 6	**route-map***name* {**permit** | **deny** | *seq-num* | *ordering-seq*}
	Example:
	`Switch(config)# route-map RM2 permit 20`
	`Switch(config-route-map)#`
		Associates ACL with a route map.
		• *name*: Specifies the route map name.
		• *seq-num*: Specifies sequence number to insert or delete from existing route map entry. The range is from 0 to 65535.
		• *ordering-seq*: Specifies ordered sequence number to insert or delete from existing route map entry.
		• **permit**: Permits set operations of the route map.
		• **deny**: Denies set operations of the route map.

Step 7	**match ip-address** *acl-name* **Example:** `Switch(config-route-map)# match ip address ACL2`	Configures the specified ACL's policies to the route map.
Step 8	**ip nat pool** *name start-ip end-ip* **netmask** *netmask* | **prefix-length** *prefix-length* **Example:** `Switch(config-route-map)# ip nat pool POOL1 5.5.5.5 5.5.5.30 prefix-length 24`	Defines a pool of global addresses to be allocated as needed.

	Command or Action	Purpose
Step 9	**ip nat inside source route-map** *name* **pool** *name* **Example:** `Switch(config-route-map)# ip nat inside source route-map RM1 pool POOL1`	Establishes dynamic rule using route map.

Configuring Rate Limit

Perform the following task to configure a limit on the dynamically created NAT entries. Packets exceeding the set limit will fail to get translated and get dropped.

Procedure

	Command or Action	Purpose
Step 1	enable **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	configure terminal **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat translation max-entries** { *number* | **all-host** *number* | **all-vrf** *number* | **host** *ip-address number* | **list** *list name number* | **vrf** *name number*} **Example:** `Device(config)# ip nat translation max-entries 300`	Configures the maximum number of NAT entries that are allowed from the specified source. • The maximum number of allowed NAT entries is 2147483647, although a typical range for a NAT rate limit is 100 to 300 entries. • When you configure a NAT rate limit for all VRF instances, each VRF instance is limited to the maximum number of NAT entries that you specify. • When you configure a NAT rate limit for a specific VRF instance, you can specify a maximum number of NAT entries for the named VRF instance that is greater than or less than that allowed for all VRF instances.
Step 4	end **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 5	show ip nat statistics **Example:** `Device# show ip nat statistics`	Optional) Displays current NAT usage information, including NAT rate limit settings. • After setting a NAT rate limit, use the **show ip nat statistics** command to check NAT rate limit related statistics.

Configuring Network Address Translation on Layer 3 Port Channel

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-number* **Example:** `Switch(config)# interface port-channel 10`	Enters port-channel interface mode.
Step 4	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for an interface.
Step 5	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 6	**interface port channel** *port-channel-number* **Example:** `Switch(config)# interface port-channel 11`	Enters port-channel interface mode.
Step 7	**ip address** *ip-address mask* **[secondary]** **Example:** `Switch(config-if)# ip address 172.31.232.182 255.255.255.240`	Sets a primary IP address for an interface.

	Command or Action	Purpose
Step 8	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 9	**end** **Example:** `Switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for Network Address Translation

Example: Configuring Static Translation of Inside Source Addresses

The following example shows how inside hosts addressed from the 10.114.11.0 network are translated to the globally unique 172.31.233.208/28 network. Further, packets from outside hosts that are addressed from the 10.114.11.0 network (the true 10.114.11.0 network) are translated to appear from the 10.0.1.0/24 network.

```
ip nat pool net-208 172.31.233.208 172.31.233.223 prefix-length 28
ip nat pool net-10 10.0.1.0 10.0.1.255 prefix-length 24
ip nat inside source list 1 pool net-208
ip nat outside source list 1 pool net-10
!
interface gigabitethernet 0/0/0
 ip address 172.31.232.182 255.255.255.240
 ip nat outside
!
interface gigabitethernet 1/1/1
 ip address 10.114.11.39 255.255.255.0
 ip nat inside
!
access-list 1 permit 10.114.11.0 0.0.0.255
```

The following example shows a static VRF aware NAT configuration to translate overlapping local addresses:

```
ip nat inside source static 192.168.121.33 10.2.2.1 vrf vrf1
ip nat inside source static 192.168.121.33.10.2.2.2 vrf vrf2
```

Example: Configuring Dynamic Translation of Inside Source Addresses

The following example shows how inside hosts addressed from either the 192.168.1.0 or the 192.168.2.0 network are translated to the globally unique 172.31.233.208/28 network:

```
ip nat pool net-208 172.31.233.208 172.31.233.223 prefix-length 9
ip nat inside source list 1 pool net-208
!
interface gigabitethernet 0/0/0
 ip address 172.31.232.182 255.255.255.240
 ip nat outside
!
interface gigabitethernet 1/1/1
 ip address 192.168.1.94 255.255.255.0
 ip nat inside
```

```
!
access-list 1 permit 192.168.1.0 0.0.0.255
access-list 1 permit 192.168.2.0 0.0.0.255
!
```

The following example shows a dynamic VRF aware NAT configuration to translate overlapping local addresses:

```
ip nat inside source list 1 interface gigabitethernet 0/0/0 vrf vrf1 overload
!
ip route vrf vrf1 0.0.0.0 0.0.0.0 192.168.1.1
!
access-list 1 permit 10.1.1.1.0 0.0.0.255
!
ip nat inside source list 1 interface gigabitethernet 1/1/1 vrf vrf1 overload
!
ip route vrf vrf1 0.0.0.0 0.0.0.0 172.16.1.1 global
access-list 1 permit 10.1.1.0 0.0.0.255
!
```

Troubleshooting NAT

This section explains the basic steps to troubleshoot and verify NAT.

- Clearly define what NAT is supposed to achieve.
- Verify that correct translation table exists using the **show ip nat translations** command.
- Verify that timer values are correctly configured using the **show ip nat translations verbose** command.
- Check the ACL values for NAT using the **show ip access-list** command
- Check the overall NAT configuration using the **show ip nat statistics** command.
- Use the **clear ip nat translations** command to clear the NAT translational table entires before the timer expires.
- Use debug nat ip and debug nat ip detailed commands to debug NAT configuration.

For further information on Troubleshooting NAT refer http://www.cisco.com/c/en/us/support/docs/ip/network-address-translation-nat/8605-13.html

Feature History for Network Address Translation

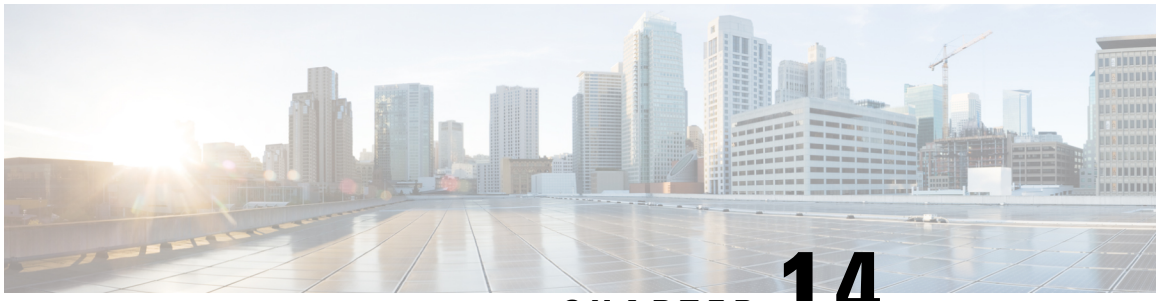
This table provides release and related information for the features explained in this module.

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 Gibraltar 16.12.1	License level for NAT	NAT is now available on the Network Advantage license. On all earlier releases, it is available on the DNA Advantage license.

Release	Feature	Feature Information
Cisco IOS XE Amsterdam 17.2.1	VRF-Aware NAT	VRF support for NAT was introduced.
Cisco IOS XE Bengaluru 17.5.1	Enhanced NAT Scale Support	Support to enhance the NAT scale using the **natscale** command in the global configuration mode was introduced.
Cisco IOS XE Bengaluru 17.6.1	Static NAT rule precedence over Dynamic NAT	This feature allows static NAT rule to precede over dynamic NAT rule when a given address is eligible for translation by both the rules.
Cisco IOS XE Cupertino 17.7.1	Network Address Translation VRF-Aware NAT	These features were implemented on supervisor modules C9400X-SUP-2 and C9400X-SUP-2XL, which were introduced in this release.
Cisco IOS XE Cupertino 17.9.1	Route Map based NAT	This feature allows you to configure route map based NAT. Route map based NAT enables destination based translation and supports match addresses. Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches.

Use the Cisco Feature Navigator to find information about platform and software image support.



CHAPTER 14

Configuring Stateful Network Address Translation 64

The Stateful NAT64 feature provides a translation mechanism that translates IPv6 packets into IPv4 packets and vice versa. This mechanism focuses on enabling connectivity across IPv6 and IPv4 domains. There are other methods like dual stacking and tunnelling but stateful NAT64 has certain advantages over them as mentioned in RFC 6144.

- Restrictions for Configuring Stateful Network Address Translation 64, on page 171
- Information About Stateful Network Address Translation 64, on page 172
- How to Configure Stateful Network Address Translation 64, on page 176
- Configuration Examples for Stateful Network Address Translation 64, on page 185
- Feature History for Configuring Stateful Network Address Translation 64, on page 187

Restrictions for Configuring Stateful Network Address Translation 64

- Applications without a corresponding application-level gateway (ALG) may not work properly with the Stateful NAT64 translator.
- IP Multicast is not supported.
- The translation of IPv4 options, IPv6 routing headers, hop-by-hop extension headers, destination option headers, and source routing headers is not supported.
- Virtual routing and forwarding-aware (VRF-aware) NAT64 is not supported.
- When traffic flows from IPv6 to IPv4, the destination IP address that you have configured must match a stateful prefix to prevent hairpinning loops. However, the source IP address (source address of the IPv6 host) must not match the stateful prefix. If the source IP address matches the stateful prefix, packets are dropped.

  Hairpinning allows two endpoints inside Network Address Translation (NAT) to communicate with each other, even when the endpoints use only each other's external IP addresses and ports for communication.
- Only TCP and UDP Layer 4 protocols are supported for header translation.
- Routemaps are not supported.

- If a static mapping host-binding entry exists for an IPv6 host, the IPv4 nodes can initiate communication. In dynamic mapping, IPv4 nodes can initiate communication only if a host-binding entry is created for the IPv6 host through a previously established connection to the same or a different IPv4 host.

  Dynamic mapping rules that use Port-Address Translation (PAT), host-binding entries cannot be created because IPv4-initiated communication not possible through PAT.

- Configuring NAT44 and NAT64 on the same interface is not recommended. Applying such a configuration could potentially impact the functionality of both NAT44 and NAT64. If such a configuration is applied, then you must remove both the configurations and re-apply the desired configuration.
- Address Only Translation is not supported.
- Post NAT fragmentation is not supported. If a packet exceeds the maximum transmission unit (MTU) after the translation, the packet will be dropped.
- NAT64 is not supported with object group-based ACLs.
- Overlapping address translation within the same VRF is not supported. You can configure only a single NAT64 rule for an IPv6 host within a given VRF. This implies that a given IPv6 host can only access IPv4 services hosted on either of the default or nondefault VRFs, but not both. However, overlapping address translation across different VRFs is supported, and NAT64 supports the translation of IPv6 hosts with overlapping addresses which are associated with different VRFs.

Note For Domain Name System (DNS) traffic to work, you must have a separate working installation of DNS64.

Information About Stateful Network Address Translation 64

Stateful Network Address Translation 64

The Stateful NAT64 feature provides a translation mechanism that translates IPv6 packets into IPv4 packets and vice versa.

Stateful NAT64 supports Internet Control Message Protocol (ICMP), TCP, and UDP traffic. Packets that are generated in an IPv6 network and are destined for an IPv4 network are routed within the IPv6 network towards the Stateful NAT64 translator. Stateful NAT64 translates the packets and forwards them as IPv4 packets through the IPv4 network. The process is reversed for traffic that is generated by hosts connected to the IPv4 network and destined for an IPv6 receiver.

The Stateful NAT64 translation is not symmetric, because the IPv6 address space is larger than the IPv4 address space and a one-to-one address mapping is not possible. Before it can perform an IPv6 to an IPv4 translation, Stateful NAT64 requires a state that binds the IPv6 address and the TCP/UDP port to the IPv4 address. The binding state is either statically configured or dynamically created when the first packet that flows from the IPv6 network to the IPv4 network is translated. After the binding state is created, packets flowing in both directions are translated. In dynamic binding, Stateful NAT64 supports communication initiated by the IPv6-only node toward an IPv4-only node. Static binding supports communication initiated by an IPv4-only node to an IPv6-only node and vice versa. Stateful NAT64 with NAT overload or Port Address Translation (PAT) provides a 1:*n* mapping between IPv4 and IPv6 addresses.

When an IPv6 node initiates traffic through Stateful NAT64, and the incoming packet does not have an existing state and the following events happen:

- The source IPv6 address (and the source port) is associated with an IPv4 configured pool address (and port, based on the configuration).
- The destination IPv6 address is translated mechanically based on RFC 7915 using either the configured NAT64 stateful prefix or the Well Known Prefix (WKP).
- The packet is translated from IPv6 to IPv4 and forwarded to the IPv4 network.

When an incoming packet is stateful (if a state exists for an incoming packet), NAT64 identifies the state and uses the state to translate the packet.

Prefix Format for Stateful Network Address Translation 64

A set of bits at the start of an IPv6 address is called the format prefix. Prefix length is a decimal value that specifies how many of the leftmost contiguous bits of an address comprise the prefix.

When packets flow from the IPv6 to the IPv4 direction, the IPv4 host address is derived from the destination IP address of the IPv6 packet that uses the prefix length. When packets flow from the IPv4 to the IPv6 direction, the IPv4 host address is constructed using the stateful prefix.

According to the IETF address format RFC 7915 a u-bit (bit 70) defined in the IPv6 architecture should be set to zero. For more information on the u-bit usage, see RFC 2464. The reserved octet, also called u-octet, is reserved for compatibility with the host identifier format defined in the IPv6 addressing architecture. RFC 6052 section 2.2 describes the address encoding/u-bit semantics. When constructing an IPv6 packet, the translator has to make sure that the u-bits are not tampered with and are set to the value suggested by RFC 2373. The suffix will be set to all zeros by the translator. IETF recommends that the 8 bits of the u-octet (bit range 64–71) be set to zero.

Well-Known Prefix

Well-Known Prefix 64:FF9B::/96 is supported for Stateful NAT64. During a stateful translation, if no stateful prefix is configured (either on the interface or globally), the WKP prefix is used to translate the IPv4 host addresses.

Performance and Scale Numbers for Stateful Nat64

Stateful NAT64 module is capable of performing translation and forwarding in the hardware at half line-rate, by programming the relevant hardware tables with the forwarding and rewrite information.

The maximum number of bidirectional translations supported on the hardware level is 3000. Packet flows that cannot be translated in the hardware will be translated-forwarded at the software level at a reduced throughput

Stateful IPv4-to-IPv6 Packet Flow

The packet flow of IPv4-initiated packets for stateful NAT64 is as follows:

- The destination address is routed to a NAT Virtual Interface (NVI).

A virtual interface is created when stateful NAT64 is configured. For stateful NAT64 translation to work, all packets must get routed to the NVI. When you configure an address pool, a route is automatically added to all IPv4 addresses in the pool. This route automatically points to the NVI.

- The IPv4-initiated packet hits static or dynamic binding.

  Static address bindings are created by the stateful NAT64 translator when you configure a static rule. Dynamic address bindings are created by stateful NAT64 translator when the IPv6-to-IPv4 traffic matches a configured dynamic non-overload rule. This creates a binding between the given IPv6 address and the corresponding IPv4 address from the associated IPv4 address pool.

- A session is created based on the translation information.

All subsequent IPv4-initiated packets are translated based on the previously created session.

Stateful NAT64 supports endpoint-dependent filtering for the IPv4-to-IPv6 packet flow with PAT configuration. In a stateful NAT64 PAT configuration, the packet flow must have originated from the IPv6 realm and created the state information in NAT64 state tables. Packets from the IPv4 side that do not have a previously created state are dropped. Endpoint-independent filtering is supported with static Network Address Translation (NAT) and non-PAT configurations.

Stateful IPv6-to-IPv4 Packet Flow

The stateful IPv6-initiated packet flow is as follows:

- The destination address is routed to a NVI.

  A virtual interface is created when stateful NAT64 is configured. A route is added to destinations that match well-known prefix (WKP) or network specific prefix (NSP) pointing to the NVI.

- A packet is considered eligible for translation when it matches a static or dynamic rule. Dynamic rule match is successful when a packet is permitted by the associated access control list. An IPv4 address and port is associated to the IPv6 destination address based on the matched rule. The IPv6 packet is translated and the IPv4 packet is formed by using the following methods:
  - Extracting the destination IPv4 address by stripping the prefix from the IPv6 address. The source address is replaced by the allocated IPv4 address (and port).
  - The rest of the fields are translated from IPv6-to-IPv4 to form a valid IPv4 packet.

- A new NAT64 translation is created in the session database and in the bind database. The pool and port databases are updated depending on the configuration. The return traffic and the subsequent traffic of the IPv6 packet flow will use this session database entry for translation.

VRF-Aware Network Address Translation 64

NAT64 is typically configured to operate across the default or the global routing domain. Translation between the IPv6 and IPv4 packets occur in the default Virtual routing and forwarding (VRF) domain. However, certain scenarios such as the BGP EVPN VXLAN fabric with Dynamic NAT64 require NAT64 to operate within a specific nondefault VRF. A common scenario would be enabling shared service access for private networks that have overlapping address space. In such a scenario, the private networks can be placed in different VRFs and access to the global service can be achieved by configuring VRF-aware NAT64 rules. These rules aid in mapping the overlapping private address to a unique global address. VRF-awareness enables NAT64 to carry

out address and port translation, and forwarding by considering the VRF in which these IPv4 and IPv6 networks exist.

Supported Deployments for VRF-Aware NAT64

In this release, VRF-aware NAT64 supports the following NAT64 translations:

- **VRF to Global NAT64 translation**:

  VRF to Global translation is between an IPv6 network that is associated with a specific user-defined VRF and an IPv4 network that is associated with the default VRF.

- **Intra-VRF NAT64 translation**:

  Intra-VRF NAT64 translation is between an IPv6 network and an IPv4 network, both within the same nondefault VRF instance. In the intra-VRF NAT64 translation, local and global addresses for the hosts are isolated to their respective VRFs, and the translated addresses may overlap each other.

  To separate the address space for the translated addresses among the VRFs, use the **match-in-vrf** keyword in the NAT64 mapping command

NAT64 Commands to Configure VRF-Awareness

The following NAT64 commands make the translations VRF-aware.

Command	Usage
nat64 prefix stateful *ipv6_prefix length* [**vrf** *vrf_name*]	Use this command in the global configuration mode to provision a VRF-aware stateful prefix configuration.
nat64 v6v4 static *ipv6_addr ipv4_addr*[**vrf** *vrf_name* [**match-in-vrf**]]	Use this command to provision a VRF-aware static rule. To enable intra-VRF translation, use the **match-in-vrf** keyword.
nat64 v6v4 list *acl_name* **pool** *pool_name* [**vrf** *vrf_name* [**match-in-vrf]**] [**overload**]	Use this command to provision a VRF-aware dynamic rule. To enable intra-VRF translation, use the **match-in-vrf** keyword. **Note** Same ACL or pool cannot be shared across different dynamic rules. However, overlapping IPv6 address translation can be achieved through different ACLs.

Related Show Commands

Use the following **show** commands to display the VRF association details:

- **show nat64 statistics**
- **show nat64 prefix stateful global**
- **show nat64 prefix stateful static-routes**

Verify the VRF-aware NAT64 translations using the following **show** commands:

- **show nat64 translations** [**vrf** *vrf_name*]
- **show nat64 statistics prefix stateful** *ipv6_prefix* [**vrf** *vrf_name*]
- **show nat64 prefix stateful static-routes prefix** *ipv6_prefix* [**vrf** *vrf_name*]

How to Configure Stateful Network Address Translation 64

Based on your network configuration, you can configure static, dynamic, or dynamic Port Address Translation (PAT) Stateful NAT64.

Note You need to configure at least one of the configurations described in the following tasks for Stateful NAT64 to work.

Configuring Static Stateful Network Address Translation 64

You can configure a static IPv6 address to an IPv4 address and vice versa. Optionally, you can configure static Stateful NAT64 with or without ports. Perform this task to configure static Stateful NAT64.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures an interface type and enters interface configuration mode.
Step 5	**description** *string* **Example:** `Device(config-if)# description interface facing ipv6`	Adds a description to an interface configuration.

	Command or Action	Purpose
Step 6	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding blue`	Associates a VRF with the IPv6 interface.
Step 7	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface.
Step 8	**ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8:1::1/96`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 9	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables NAT64 translation on an IPv6 interface.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 11	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/2/0`	Configures an interface and enters interface configuration mode.
Step 12	**description** *string* **Example:** `Device(config-if)# description interface facing ipv4`	Adds a description to an interface configuration.
Step 13	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 209.165.201.1 255.255.255.0`	Configures an IPv4 address for an interface.
Step 14	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables NAT64 translation on an IPv4 interface.
Step 15	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.

	Command or Action	Purpose
Step 16	**nat64 prefix stateful** *ipv6-prefix/length* [**vrf** *vrf-name*] **Example:** `Device(config)# nat64 prefix stateful 2001:DB8:1::1/96 vrf blue`	Defines the Stateful NAT64 prefix to be added to IPv4 hosts to translate the IPv4 address into an IPv6 address. The Stateful NAT64 prefix can be configured at the global configuration mode or at the interface mode. Use the **vrf** keyword in the global configuration mode to provision a VRF-aware stateful prefix configuration.
Step 17	**nat64 v6v4 static** *ipv6-address ipv4-address* [**vrf** *vrf-name*] **Example:** `Device(config)# nat64 v6v4 static 2001:DB8:1::FFFE 209.165.201.1 vrf blue`	Enables NAT64 IPv6-to-IPv4 static address mapping.
Step 18	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring Dynamic Stateful Network Address Translation 64

A dynamic Stateful NAT64 configuration provides a one-to-one mapping of IPv6 addresses to IPv4 addresses in the address pool. You can use the dynamic Stateful NAT64 configuration when the number of active IPv6 hosts is less than the number of IPv4 addresses in the pool. Perform this task to configure dynamic Stateful NAT64.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:**	Configures an interface type and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface gigabitethernet 0/0/0`	
Step 5	**description** *string* **Example:** `Device(config-if)# description interface facing ipv6`	Adds a description to an interface configuration.
Step 6	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding blue`	Associates a VRF with the IPv6 interface.
Step 7	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface.
Step 8	**ipv6** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*} **Example:** `Device(config-if)# ipv6 2001:DB8:1::1/96`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 9	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables Stateful NAT64 translation on an IPv6 interface.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 11	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/2/0`	Configures an interface type and enters interface configuration mode
Step 12	**description** *string* **Example:** `Device(config-if)# description interface facing ipv4`	Adds a description to an interface configuration.
Step 13	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 209.165.201.24 255.255.255.0`	Configures an IPv4 address for an interface.
Step 14	**nat64 enable** **Example:**	Enables Stateful NAT64 translation on an IPv4 interface.

	Command or Action	Purpose
	`Device(config-if)# nat64 enable`	
Step 15	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 16	**ipv6 access-list** *access-list-name* **Example:** `Device(config)# ipv6 access-list nat64-acl`	Defines an IPv6 access list and enters IPv6 access list configuration mode.
Step 17	**permit ipv6** *ipv6-address* **any** **Example:** `Device(config-ipv6-acl)# permit ipv6 2001:DB8:2::/96 any`	Sets permit conditions for an IPv6 access list.
Step 18	**exit** **Example:** `Device(config-ipv6-acl# exit`	Exits IPv6 access list configuration mode and enters global configuration mode.
Step 19	**nat64 prefix stateful** *ipv6-prefix/length* [**vrf** *vrf-name*] **Example:** `Device(config)# nat64 prefix stateful 2001:DB8:1::1/96 vrf blue`	Enables NAT64 IPv6-to-IPv4 address mapping. Use the **vrf** keyword to provision a VRF-aware stateful prefix configuration.
Step 20	**nat64 v4 pool** *pool-name start-ip-address end-ip-address* **Example:** `Device(config)# nat64 v4 pool pool1 209.165.201.1 209.165.201.254`	Defines the Stateful NAT64 IPv4 address pool.
Step 21	**nat64 v6v4 list** *access-list-name* **pool** *pool-name* [**vrf** *vrf-name* [**match-in-vrf**]] **Example:** `Device(config)# nat64 v6v4 list nat64-acl pool pool1 vrf blue`	Dynamically translates an IPv6 source address to an IPv4 source address and an IPv6 destination address to an IPv4 destination address for NAT64. Use the **vrf** keyword to provision a VRF-aware translation. Use the **match-in-vrf** keyword to enable intra-VRF translation.
Step 22	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring Dynamic Port Address Translation Stateful NAT64

A Port Address Translation (PAT) or overload configuration is used to multiplex (mapping IPv6 addresses to a single IPv4 pool address) multiple IPv6 hosts to a pool of available IPv4 addresses on a first-come first-served basis. The dynamic PAT configuration conserves the IPv4 address space while providing connectivity to the IPv4 Internet. Configure the **nat64 v6v4 list** command with the **overload** keyword to configure PAT address translation. Perform this task to configure dynamic PAT Stateful NAT64.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures an interface type and enters interface configuration mode.
Step 5	**description** *string* **Example:** `Device(config-if)# description interface facing ipv6`	Adds a description to an interface configuration.
Step 6	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding blue`	Associates a VRF with the IPv6 interface.
Step 7	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface.
Step 8	**ipv6** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*} **Example:** `Device(config-if)# ipv6 2001:DB8:1::1/96`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.

	Command or Action	Purpose
Step 9	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables Stateful NAT64 translation on an IPv6 interface.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 11	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/2/0`	Configures an interface type and enters interface configuration mode
Step 12	**description** *string* **Example:** `Device(config-if)# description interface facing ipv4`	Adds a description to an interface configuration.
Step 13	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 209.165.201.24 255.255.255.0`	Configures an IPv4 address for an interface.
Step 14	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables Stateful NAT64 translation on an IPv6 interface.
Step 15	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 16	**ipv6 access-list** *access-list-name* **Example:** `Device(config)# ipv6 access-list nat64-acl`	Defines an IPv6 access list and places the device in IPv6 access list configuration mode.
Step 17	**permit ipv6** *ipv6-address* **any** **Example:** `Device(config-ipv6-acl)# permit ipv6 2001:db8:2::/96 any`	Sets permit conditions for an IPv6 access list.
Step 18	**exit** **Example:** `Device(config-ipv6-acl)# exit`	Exits IPv6 access list configuration mode and enters global configuration mode.

	Command or Action	Purpose
Step 19	**nat64 prefix stateful** *ipv6-prefix/length* [**vrf** *vrf-name*] **Example:** `Device(config)# nat64 prefix stateful 2001:db8:1::1/96 vrf blue`	Enables NAT64 IPv6-to-IPv4 address mapping.
Step 20	**nat64 v4 pool** *pool-name start-ip-address end-ip-address* **Example:** `Device(config)# nat64 v4 pool pool1 209.165.201.1 209.165.201.254`	Defines the Stateful NAT64 IPv4 address pool.
Step 21	**nat64 v6v4 list** *access-list-name* **pool** *pool-name* [**vrf** *vrf-name* [**match-in-vrf**]] **overload** **Example:** `Device(config)# nat64 v6v4 list nat64-acl pool pool1 vrf blue overload`	Enables NAT64 PAT or overload address translation. Use the **vrf** keyword to provision a VRF-aware configuration. Use the **match-in-vrf** option to configure intra-VRF NAT64 PAT.
Step 22	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring Timeout Functionality

Before you begin

Perform this task to configure idle timeout for dynamically created NAT64 session and bind entries.

Procedure

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

Step 2	**configure terminal**
	Example:
	`Device# configure terminal`
	`Device(config)#`
		Enters global configuration mode.

Step 3	**nat64 translation timeout** { **bind** *time* | **icmp** *time*|**tcp** *time*|**tcp-transient** *time*|**udp** *time* } **Example:**	Configures timeout functionality for binds and session entries.

	Command or Action	Purpose
	`Device(config)#nat64 translation timeout bind 2`	• **bind** *time*: Specifies the timeout for NAT64 binds. The default timeout value is 1 hour. You can change the timeout value using the *time* option. • **icmp** *time*: Specifies the timeout for NAT64 ICMP traffic flow. The default timout value is 60 seconds. You can change the timeout value using the *time* option. • **tcp** *time*: Specifies the timeout for NAT64 TCP traffic flow. The default timeout value is 2 hours. You can change the timeout value using the *time* option. • **tcp-transient** *time*: Specifies the timeout for NAT64 transient TCP traffic flow. The default timeout value is 4 minutes. You can change the timeout value using the *time* option. • **udp** *time*: Specifies the timeout for NAT64 UDP traffic flow. The default timeout value is 5 minutes. You can change the timeout value using the *time* option.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring Switch Database Management Template

Use (SDM) templates to configure system resources to optimize support for stateful NAT64.

After you set 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 output displays the template currently in use and the template that will become active after a reload.

Follow these steps to set the SDM template to maximize stateful NAT64 usage:

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	**sdm prefer custom acl** **Example:** `Device(config)#sdm prefer custom acl`	Creates a customizable SDM template for ACL features. Enters a submode for customizing features.
Step 4	**pbr** *number-of-entries* **priority** *priority-value* **Example:** `Device(config-sdm-acl)#pbr 27 priority 1`	Specifies the number of entries allotted for PBR or NAT. The value ranges from 2 K to 27 K. The value is rounded up to the next 2 K unit. The priority values range from 1 to 8.
Step 5	**sdm prefer custom commit** **Example:** `Device(config)# sdm prefer custom commit`	Changes the running SDM preferences to the values in the customized template. The new template takes effect in the next reload.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**reload** **Example:** `Device# reload`	Reloads the device and applies the customized SDM template.

Configuration Examples for Stateful Network Address Translation 64

Example: Configuring Static Stateful Network Address Translation 64

```
Device# configure terminal
Device(config)# ipv6 unicast-routing
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# description interface facing ipv6
Device(config-if)# ipv6 enable
```

```
Device(config-if)# ipv6 address 2001:DB8:1::1/96
Device(confif-if)# nat64 enable
Device(config-fi)# exit
Device(config)# interface gigabitethernet 1/2/0
Device(config-if)# description interface facing ipv4
Device(config-if)# ip address 209.165.201.1 255.255.255.0
Device(config-if)# nat64 enable
Device(config-if)# exit
Device(config)# nat64 prefix stateful 2001:DB8:1::1/96
Device(config)# nat64 v6v4 static 2001:DB8:1::FFFE 209.165.201.1
Device(config)# end
```

Example: Configuring Dynamic Stateful Network Address Translation 64

The following example shows how to configure dynamic Stateful NAT64 with intra-VRF translation enabled for the VRF blue.

```
Device# configure terminal
Device(config)# ipv6 unicast-routing
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# description interface facing ipv6
Device(config-if)# vrf forwarding blue
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 2001:DB8:1::1/96
Device(config-if)# nat64 enable
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/2/0
Device(config-if)# description interface facing ipv4
Device(config-if)# vrf forwarding blue
Device(config-if)# ip address 209.165.201.24 255.255.255.0
Device(config-if)# nat64 enable
Device(config-if)# exit
Device(config)# ipv6 access-list nat64-acl
Device(config-ipv6-acl)# permit ipv6 2001:db8:2::/96 any
Device(config-ipv6-acl)# exit
Device(config)# nat64 prefix stateful 2001:db8:1::1/96 vrf blue
Device(config)# nat64 v4 pool pool1 209.165.201.1 209.165.201.254
Device(config)# nat64 v6v4 list nat64-acl pool pool1 vrf blue match-in-vrf
Device(config)# end
```

Example: Configuring Dynamic Port Address Translation Stateful NAT64

```
enable
 configure terminal
  ipv6 unicast-routing
  interface gigabitethernet 0/0/0
   description interface facing ipv6
   ipv6 enable
   ipv6 2001:DB8:1::1/96
   nat64 enable
   exit
  interface gigabitethernet 1/2/0
   description interface facing ipv4
   ip address 209.165.201.24 255.255.255.0
   nat64 enable
   exit
```

```
ipv6 access-list nat64-acl
 permit ipv6 2001:db8:2::/96 any
 exit
nat64 prefix stateful 2001:db8:1::1/96
nat64 v4 pool pool1 209.165.201.1 209.165.201.254
nat64 v6v4 list nat64-acl pool pool1 overload
end
```

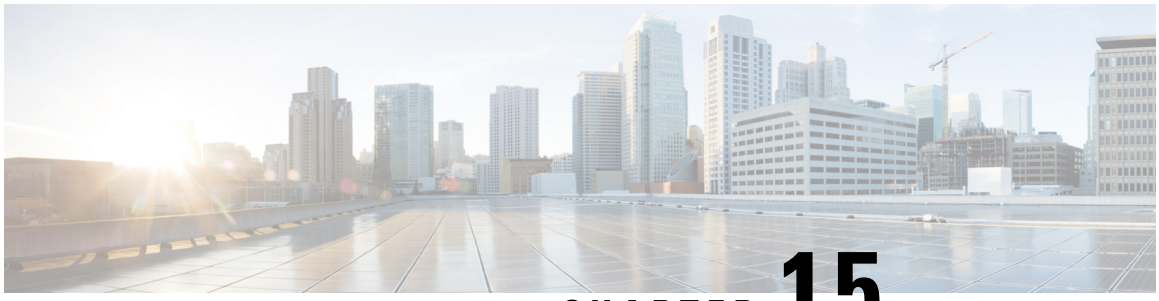
Feature History for Configuring Stateful Network Address Translation 64

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 Dublin 17.10.1	Configuring Stateful Network Address Translation 64	The Stateful NAT64 feature provides a translation mechanism that translates IPv6 packets into IPv4 packets and vice versa. Packets that are generated in an IPv6 network and are destined for an IPv4 network are routed within the IPv6 network towards the Stateful NAT64 translator. Stateful NAT64 translates the packets and forwards them as IPv4 packets through the IPv4 network. The process is reversed for traffic that is generated by hosts connected to the IPv4 network and destined for an IPv6 receiver.
Cisco IOS XE Dublin 17.12.1	VRF-aware Stateful Network Address Translation 64	NAT64 is made VRF-aware, and it supports the translation in the user-defined VRFs. The following translation scenarios are supported: • VRF to Global: IPv6 device is in a user-defined VRF and it communicates with an IPv4 device that is default VRF. • Intra-VRF: An IPv6 device communicates with an IPv4 device that is in the same user-defined VRF.

Use the Cisco Feature Navigator to find information about platform and software image support.



CHAPTER 15

VRRPv3 Protocol Support

- Restrictions for VRRPv3 Protocol Support, on page 189
- Information About VRRPv3 Protocol Support, on page 190
- How to Configure VRRPv3 Protocol Support, on page 192
- Configuration Examples for VRRPv3 Protocol Support, on page 195
- Additional References, on page 197
- Feature History for VRRPv3 Protocol Support, on page 197

Restrictions for VRRPv3 Protocol Support

- VRRPv3 is not intended as a replacement for existing dynamic protocols. VRRPv3 is designed for use over multi-access, multicast, or broadcast capable Ethernet LANs.
- For Cisco Catalyst 9600X Series Supervisor 2 Module, VRRP is supported only on the switch virtual interface (SVI) and not on any other routed interfaces. For Cisco Catalyst 9500 Series Switches and Cisco Catalyst 9500 Series - High Performance, VRRPv3 is supported on Ethernet, Fast Ethernet, Bridge Group Virtual Interface (BVI), and Gigabit Ethernet interfaces, and on Multiprotocol Label Switching (MPLS) Virtual Private Networks (VPNs), VRF-aware MPLS VPNs and VLANs.
- Because of the forwarding delay that is associated with the initialization of a BVI interface, you must not configure the VRRPv3 advertise timer to a value lesser than the forwarding delay on the BVI interface. If you configure the VRRPv3 advertise timer to a value equal to or greater than the forwarding delay on the BVI interface, the setting prevents a VRRP device on a recently initialized BVI interface from unconditionally taking over the primary role. Use the **bridge forward-time** command to set the forwarding delay on the BVI interface. Use the **vrrp timers advertise** command to set the VRRP advertisement timer.
- VRRPv3 does not support Stateful Switchover (SSO).
- Full network redundancy can only be achieved if VRRP operates over the same network path as the VRRS Pathway redundant interfaces. For full redundancy, the following restrictions apply:
  - VRRS pathways should not share a different physical interface as the parent VRRP group or be configured on a sub-interface having a different physical interface as the parent VRRP group.
  - VRRS pathways should not be configured on Switch Virtual Interface (SVI) interfaces as long as the associated VLAN does not share the same trunk as the VLAN on which the parent VRRP group is configured.

- Interface link-local IP address and VRRP group virtual link-local IP address should be different for VRRP features to work properly.

Information About VRRPv3 Protocol Support

The following sections provide information about VRRPv3 protocol support.

VRRPv3 Benefits

Support for IPv4 and IPv6

VRRPv3 supports IPv4 and IPv6 address families while VRRPv2 only supports IPv4 addresses.

Note When VRRPv3 is in use, VRRPv2 is unavailable. For VRRPv3 to be configurable, the **fhrp version vrrp v3** command must be used in global configuration mode

Redundancy

VRRP enables you to configure multiple devices as the default gateway device, which reduces the possibility of a single point of failure in a network.

Load Sharing

You can configure VRRP in such a way that traffic to and from LAN clients can be shared by multiple devices, thereby sharing the traffic load more equitably between available devices.

Multiple Virtual Devices

VRRP supports up to 255 virtual devices (VRRP groups) on a device physical interface, subject to restrictions in scaling. Multiple virtual device support enables you to implement redundancy and load sharing in your LAN topology. In scaled environments, VRRS Pathways should be used in combination with VRRP control groups.

Multiple IP Addresses

The virtual device can manage multiple IP addresses, including secondary IP addresses. Therefore, if you have multiple subnets configured on an Ethernet interface, you can configure VRRP on each subnet.

Note To utilize secondary IP addresses in a VRRP group, a primary address must be configured on the same group.

Preemption

The redundancy scheme of VRRP enables you to preempt a virtual device backup that has taken over for a failing primary virtual device with a higher priority virtual device backup that has become available.

Note Preemption of a lower priority primary device is enabled with an optional delay.

Advertisement Protocol

VRRP uses a dedicated Internet Assigned Numbers Authority (IANA) standard multicast address for VRRP advertisements. For IPv4, the multicast address is 224.0.0.18. For IPv6, the multicast address is FF02:0:0:0:0:0:0:12. This addressing scheme minimizes the number of devices that must service the multicasts and allows test equipment to accurately identify VRRP packets on a segment. The IANA has assigned VRRP the IP protocol number 112.

Suport for SSO

Beginning from Cisco IOS XE Bengaluru 17.6.1, VRRPv3 supports Stateful Switchover (SSO). For VRRPv3 to support SSO, the **fhrp sso** command should be enabled. You can disable SSO support using the **no fhrp sso** command.

VRRP Device Priority and Preemption

An important aspect of the VRRP redundancy scheme is VRRP device priority. Priority determines the role that each VRRP device plays and what happens if the primary virtual device fails.

If a VRRP device owns the IP address of the virtual device and the IP address of the physical interface, this device will function as a primary virtual device.

Priority also determines if a VRRP device functions as a virtual device backup and the order of ascendancy to becoming a primary virtual device if the primary virtual device fails. You can configure the priority of each virtual device backup with a value of 1 through 254 using the **priority** command (use the **vrrp address-family** command to enter the VRRP configuration mode and access the **priority** option).

For example, if device A, the primary virtual device in a LAN topology, fails, an election process takes place to determine if virtual device backups B or C should take over. If devices B and C are configured with the priorities of 101 and 100, respectively, device B is elected to become primary virtual device because it has the higher priority. If devices B and C are both configured with the priority of 100, the virtual device backup with the higher IP address is elected to become the primary virtual device.

By default, a preemptive scheme is enabled whereby a higher priority virtual device backup that becomes available takes over from the virtual device backup that was elected to become primary virtual device. You can disable this preemptive scheme using the **no preempt** command (use the **vrrp address-family** command to enter the VRRP configuration mode, and enter the **no preempt** command). If preemption is disabled, the virtual device backup that is elected to become primary virtual device remains the primary until the original primary virtual device recovers and becomes primary again.

Note Preemption of a lower priority primary device is enabled with an optional delay.

VRRP Advertisements

The primary virtual device sends VRRP advertisements to other VRRP devices in the same group. The advertisements communicate the priority and state of the primary virtual device. The VRRP advertisements are encapsulated into either IPv4 or IPv6 packets (based on the VRRP group configuration) and sent to the appropriate multicast address assigned to the VRRP group. For IPv4, the multicast address is 224.0.0.18. For IPv6, the multicast address is FF02:0:0:0:0:0:0:12. The advertisements are sent every second by default and the interval is configurable.

Cisco devices allow you to configure millisecond timers, which is a change from VRRPv2. You need to manually configure the millisecond timer values on both the primary and the backup devices. The primary advertisement value displayed in the **show vrrp** command output on the backup devices is always 1 second because the packets on the backup devices do not accept millisecond values.

You must use millisecond timers where absolutely necessary and with careful consideration and testing. Millisecond values work only under favorable circumstances. The use of the millisecond timer values is compatible with third party vendors, as long as they also support VRRPv3. You can specify a timer value between 100 milliseconds and 40000 milliseconds.

How to Configure VRRPv3 Protocol Support

The following sections provide configuration information about VRRPv3 protocol support.

Creating and Customizing a VRRP Group

To create a VRRP group, perform the following task. Steps 6 to 14 denote customizing options for the group, and they are optional:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**fhrp version vrrp v3** **Example:** `Device(config)# fhrp version vrrp v3`	Enables the ability to configure VRRPv3 and VRRS.
Step 4	**interface** *type number* **Example:**	Enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface GigabitEthernet 0/0/0`	
Step 5	**vrrp** *group-id* **address-family** {**ipv4** | **ipv6**} **Example:** `Device(config-if)# vrrp 3 address-family ipv4`	Creates a VRRP group and enters VRRP configuration mode.
Step 6	**address** *ip-address* [**primary** | **secondary**] **Example:** `Device(config-if-vrrp)# address 100.0.1.10 primary`	Specifies a primary or secondary address for the VRRP group. **Note** VRRPv3 for IPv6 requires that a primary virtual link-local IPv6 address is configured to allow the group to operate. After the primary link-local IPv6 address is established on the group, you can add the secondary global addresses.
Step 7	**description** *group-description* **Example:** `Device(config-if-vrrp)# description group 3`	(Optional) Specifies a description for the VRRP group.
Step 8	**match-address** **Example:** `Device(config-if-vrrp)# match-address`	(Optional) Matches secondary address in the advertisement packet against the configured address. **Note** Secondary address matching is enabled by default.
Step 9	**preempt delay minimum** *seconds* **Example:** `Device(config-if-vrrp)# preempt delay minimum 30`	(Optional) Enables preemption of lower priority primary device with an optional delay. **Note** Preemption is enabled by default.
Step 10	**priority** *priority-level* **Example:** `Device(config-if-vrrp)# priority 3`	(Optional) Specifies the priority value of the VRRP group. The priority of a VRRP group is 100 by default.
Step 11	**timers advertise** *interval* **Example:** `Device(config-if-vrrp)# timers advertise 1000`	(Optional) Sets the advertisement timer in milliseconds. The advertisement timer is set to 1000 milliseconds by default.

	Command or Action	Purpose
Step 12	**vrrpv2** **Example:** `Device(config-if-vrrp)# vrrpv2`	(Optional) Enables support for VRRPv2 configured devices in compatibility mode.
Step 13	**vrrs leader** *vrrs-leader-name* **Example:** `Device(config-if-vrrp)# vrrs leader leader-1`	(Optional) Specifies a leader's name to be registered with VRRS and to be used by followers. **Note** A registered VRRS name is unavailable by default.
Step 14	**shutdown** **Example:** `Device(config-if-vrrp)# shutdown`	(Optional) Disables VRRP configuration for the VRRP group. **Note** VRRP configuration is enabled for a VRRP group by default.
Step 15	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Delay Period Before FHRP Client Initialization

To configure the delay period before the initialization of all FHRP clients on an interface, perform the following task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**fhrp version vrrp v3** **Example:**	Enables the ability to configure VRRPv3 and VRRS.

	Command or Action	Purpose
	`Device(config)# fhrp version vrrp v3`	
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Enters interface configuration mode.
Step 5	**fhrp delay** {[**minimum**] [**reload**] *seconds*} **Example:** `Device(config-if)# fhrp delay minimum 5`	Specifies the delay period for the initialization of FHRP clients after an interface comes up. The range is 0-3600 seconds.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuration Examples for VRRPv3 Protocol Support

The following sections provide configuration examples for VRRPv3 protocol support.

Example: Enabling VRRPv3 on a Device

The following example shows how to enable VRRPv3 on a device:

```
Device> enable
Device# configure terminal
Device(config)# fhrp version vrrp v3
Device(config-if-vrrp)# end
```

Example: Creating and Customizing a VRRP Group

The following example shows how to create and customize a VRRP group:

```
Device> enable
Device# configure terminal
Device(config)# fhrp version vrrp v3
Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# vrrp 3 address-family ipv4
Device(config-if-vrrp)# address 100.0.1.10 primary
Device(config-if-vrrp)# description group 3
Device(config-if-vrrp)# match-address
Device(config-if-vrrp)# preempt delay minimum 30
Device(config-if-vrrp)# end
```

Note In the above example, the **fhrp version vrrp v3** command is used in the global configuration mode.

Example: Configuring the Delay Period Before FHRP Client Initialization

The following example shows how to configure the delay period before FHRP client initialization :

```
Device> enable
Device# configure terminal
Device(config)# fhrp version vrrp v3
Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# fhrp delay minimum 5
Device(config-if-vrrp)# end
```

Note In the above example, a five-second delay period is specified for the initialization of FHRP clients after the interface comes up. You can specify a delay period between 0 and 3600 seconds.

Example: VRRP Status, Configuration, and Statistics Details

The following is a sample output of the status, configuration, and statistics details for a VRRP group:

```
Device> enable
Device# show vrrp detail

 GigabitEthernet1/0/1 - Group 3 - Address-Family IPv4
  Description is "group 3"
  State is MASTER
  State duration 53.901 secs
  Virtual IP address is 100.0.1.10
  Virtual MAC address is 0000.5E00.0103
  Advertisement interval is 1000 msec
  Preemption enabled, delay min 30 secs (0 msec remaining)
  Priority is 100
  Master Router is 10.21.0.1 (local), priority is 100
  Master Advertisement interval is 1000 msec (expires in 832 msec)
  Master Down interval is unknown
  VRRPv3 Advertisements: sent 61 (errors 0) - rcvd 0
  VRRPv2 Advertisements: sent 0 (errors 0) - rcvd 0
  Group Discarded Packets: 0
    VRRPv2 incompatibility: 0
    IP Address Owner conflicts: 0
    Invalid address count: 0
    IP address configuration mismatch : 0
    Invalid Advert Interval: 0
    Adverts received in Init state: 0
    Invalid group other reason: 0
  Group State transition:
    Init to master: 0
    Init to backup: 1 (Last change Sun Mar 13 19:52:56.874)
    Backup to master: 1 (Last change Sun Mar 13 19:53:00.484)
    Master to backup: 0
```

```
    Master to init: 0
    Backup to init: 0

Device# exit
```

Additional References

Related Documents

Related Topic	Document Title
FHRP commands	First Hop Redundancy Protocols Command Reference
Configuring VRRPv2	*Configuring VRRP*
VRRPv3 Commands	For complete syntax and usage information for the commands used in this chapter. See *Command Reference (Catalyst 9600 Series Switches)*

Standards and RFCs

Standard/RFC	Title
RFC5798	*Virtual Router Redundancy Protocol*

Feature History for VRRPv3 Protocol Support

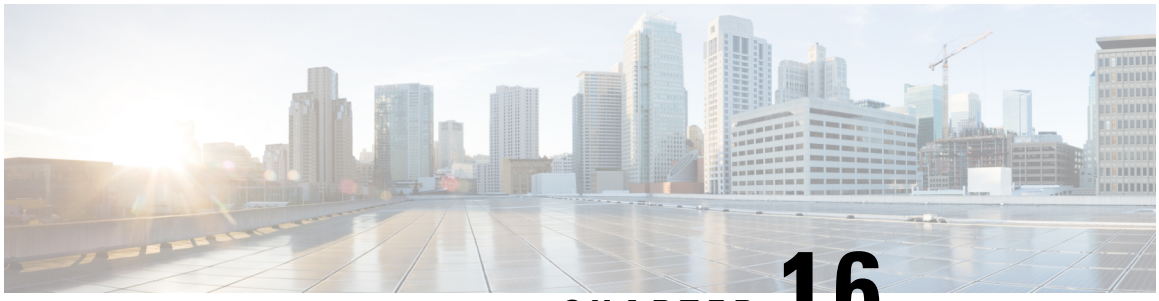
This table provides release and related information for the features explained in this module.

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 Bengaluru 17.6.1	SSO Support for VRRPv3	Beginning from 17.6.1, VRRPv3 supports Stateful Switchover (SSO). For VRRPv3 to support SSO, the **fhrp sso** command should be enabled. SSO can be disabled using the **no fhrp sso** command.

Release	Feature	Feature Information
Cisco IOS XE Gibraltar 16.11.1	VRRPv3 Protocol Support	VRRP enables a group of devices to form a single virtual device to provide redundancy. The LAN clients can then be configured with the virtual device as their default gateway. The virtual device, representing a group of devices, is also known as a VRRP group. The VRRPv3 Protocol Support feature provides the capability to support IPv4 and IPv6 addresses.
Cisco IOS XE Cupertino 17.7.1	VRRPv3 Protocol Support	This feature was implemented on the Cisco Catalyst 9600 Series Supervisor 2 Module (C9600X-SUP-2).

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/



CHAPTER 16

Configuring WCCP

This section provides information about configuring WCCP.

- Prerequisites for WCCP, on page 199
- Restrictions for WCCP, on page 199
- Information About WCCP, on page 200
- How to Configure WCCP, on page 207
- Configuration Examples for WCCP, on page 214
- Feature History for WCCP, on page 219

Prerequisites for WCCP

- To use WCCP, IP must be configured on the interface connected to the Internet and another interface must be connected to the content engine.
- The interface connected to the content engine must be a Fast Ethernet or Gigabit Ethernet interface.

Restrictions for WCCP

General

The following limitations apply to Web Cache Communication Protocol Version 2 (WCCPv2):

- WCCP works only with IPv4 networks.
- WCCP is not supported with IPv4 and IPv6 networks on Cisco Catalyst 9600X Series Switches.
- WCCP bypasses Network Address Translation (NAT) when Cisco Express Forwarding is enabled.
- WCCP does not interoperate with NAT and the zone-based firewall configured together in a network.
- Service groups can comprise up to 32 content engines and 32 switches.
- For switches servicing a multicast cluster, the Time To Live (TTL) value must be set at 15 or fewer.
- All content engines in a cluster must be configured to communicate with all devices servicing the cluster.
- Multicast addresses must be from 224.0.0.0 to 239.255.255.255.

- Up to eight service groups are supported at the same time on the same client interface.
- The Layer 2 rewrite forwarding method is supported; but generic routing encapsulation (GRE) is not.
- Direct Layer 2 connectivity to content engines is required when Layer 2 mode is deployed; Layer 3 connectivity of one or more hops away is not supported.
- Ternary content addressable memory (TCAM) friendly mask-based assignment is supported, but the hash bucket-based method is not.
- When the TCAM space is exhausted, traffic is not redirected but is forwarded normally.
- The WCCP version 2 standard allows for support of up to 256 distinct masks. However, a Cisco Catalyst 9000 series switch supports only mask assignment tables with a single mask.
- A content engine configured for mask assignment that tries to join a farm where the selected assignment method is hash remains out of the farm as long as the cache engine assignment method does not match that of the existing farm.
- WCCP redirection is not supported on Multiprotocol Label Switching (MPLS) and port-channel interfaces.
- WCCP high availability is not supported in modular, stacking, and StackWise Virtual (SVL) mode.
- This feature is not supported on the Cisco Catalyst 9600 Series Supervisor 2 Module (C9600X-SUP-2).
- The **packets redirected** counter in the output of the command **show ip wccp <*service_group*> detail** only increments whenever a packet is redirected via CPU switching. Whenever Cisco Express Forwarding (CEF) is being used with inbound WCCP redirection, L2 forwarding method, and mask assignment, all WCCP traffic should be redirected via hardware. When traffic is redirected via hardware the counters will not increment.

Catalyst 9000 Series Switches Access Control Lists

When WCCP is using the mask assignment, any redirect list is merged with the mask information from the appliance and the resulting merged ACL is passed down to the Catalyst 9000 series switch hardware. Only Permit or Deny ACL entries from the redirect list in which the protocol is IP or exactly matches the service group protocol are merged with the mask information from the appliance.

The following restrictions apply to the redirect-list ACL:

- The ACL must be an IPv4 extended ACL.
- The only valid matching criteria are **dscp** and **tos**.
- The use of **fragments**, **time-range**, or **options** keywords, or any TCP flags is not permitted.
- If the redirect ACL does not meet the restrictions shown, the system will log the following error message:

```
WCCP-3-BADACE: Service <service group>, invalid access-list entry (seq:<sequence>,
reason:<reason>)
```

Information About WCCP

The Web Cache Communication Protocol (WCCP) is a Cisco-developed content-routing technology that intercepts IP packets and redirects those packets to a destination other than the one specified in the IP packet.

Typically the packets are redirected from their destination web server on the Internet to a content engine that is local to the client. In some WCCP deployment scenarios, redirection of traffic may also be required from the web server to the client. WCCP enables you to integrate content engines into your network infrastructure.

The tasks in this document assume that you have already configured content engines on your network.

WCCP Overview

WCCP uses Cisco Content Engines (or other content engines running WCCP) to localize traffic patterns in the network, enabling content requests to be fulfilled locally. Traffic localization reduces transmission costs and download time.

WCCP enables Cisco IOS XE platforms to transparently redirect content requests. With transparent redirection, users can fulfill content requests locally without configuring their browsers to use a web proxy. Instead, they can use the target URL to request content, and have their requests automatically redirected to a content engine. The word "transparent" in this case means that the end user does not know that a requested file (such as a web page) came from the content engine instead of from the originally specified server.

A content engine receiving a request attempts to service it from its own local cache. If the requested information is not present, the content engine issues its own request to the originally targeted server to get the required information. A content engine retrieving the requested information forwards it to the requesting client and caches it to fulfill future requests, thus maximizing download performance and substantially reducing transmission costs.

WCCP enables a series of content engines, called a content engine cluster, to provide content to a device or multiple devices. Network administrators can easily scale their content engines to manage heavy traffic loads through these clustering capabilities. Cisco clustering technology enables each cluster member to work in parallel, resulting in linear scalability. Clustering content engines greatly improves the scalability, redundancy, and availability of your caching solution. You can cluster up to 32 content engines to scale to your desired capacity.

WCCP Mask Assignment

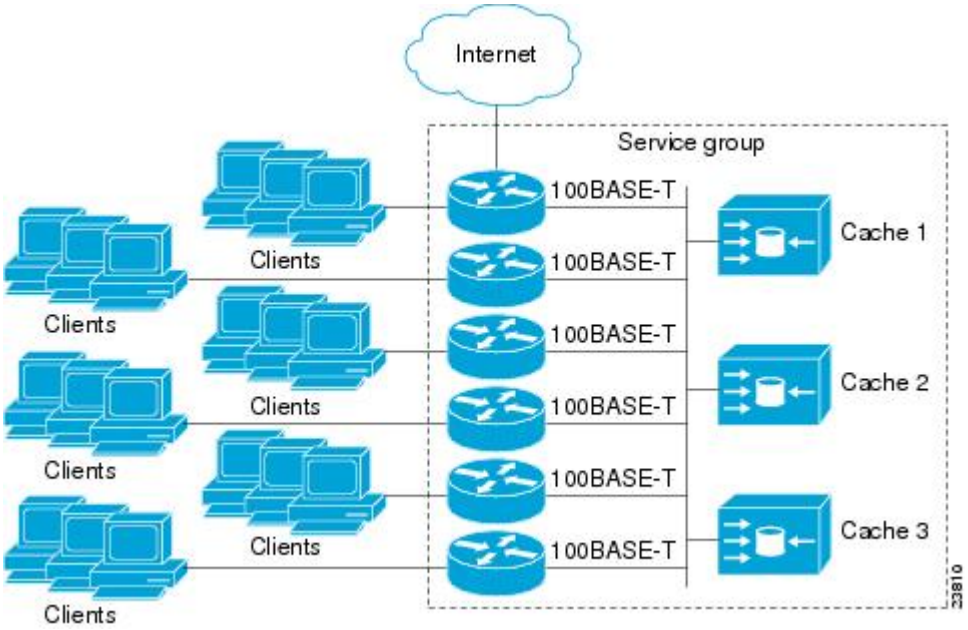
The WCCP Mask Assignment feature enables mask assignment as the load-balancing method (instead of the default hash assignment method) for a WCCP service.

For content engines running Application and Content Networking System (ACNS) software, use the **wccp custom-web-cache** command with the **mask-assign** keyword to configure mask assignment. For content engines running Cisco Wide Area Application Services (WAAS) software, use the **wccp tcp-promiscuous** command with the **mask-assign** keyword to configure mask assignment.

WCCPv2 Configuration

Multiple devices can use WCCPv2 to service a content engine cluster. The figure below illustrates a sample configuration using multiple devices.

Figure 20: Cisco Content Engine Network Configuration Using WCCPv2



The subset of content engines within a cluster and devices connected to the cluster that are running the same service is known as a service group. Available services include TCP and UDP redirection.

WCCPv2 requires that each content engine be aware of all the devices in the service group. To specify the addresses of all the devices in a service group, choose one of the following methods:

- Unicast—A list of device addresses for each of the devices in the group is configured on each content engine. In this case, the address of each device in the group must be explicitly specified for each content engine during configuration.
- Multicast—A single multicast address is configured on each content engine. In the multicast address method, the content engine sends a single-address notification that provides coverage for all switches in the service group. For example, a content engine could indicate that packets should be sent to a multicast address of 224.0.0.100, which would send a multicast packet to all devices in the service group configured for group listening using WCCP (see the **ip wccp group-listen** interface configuration command for details).

The multicast option is easier to configure because you need only specify a single address on each content engine. This option also allows you to add and remove routers from a service group dynamically, without needing to reconfigure the content engines with a different list of addresses each time.

The following sequence of events details how WCCPv2 configuration works:

1. Each content engine is configured with a list of devices.
2. Each content engine announces its presence and a list of all devices with which it has established communications. The routers reply with their view (list) of content engines in the group.
3. When the view is consistent across all content engines in the cluster, one content engine is designated as the lead and sets the policy that the devices need to deploy in redirecting packets.

WCCPv2 Support for Services Other than HTTP

WCCPv2 allows redirection of traffic other than HTTP (TCP port 80 traffic), including a variety of UDP and TCP traffic. WCCPv2 supports the redirection of packets intended for other ports, including those used for proxy-web cache handling, File Transfer Protocol (FTP) caching, FTP proxy handling, web caching for ports other than 80, and Real Audio, video, and telephony applications.

To accommodate the various types of services available, WCCPv2 introduced the concept of multiple *service groups*. Service information is specified in the WCCP configuration commands using dynamic services identification numbers (such as 98) or a predefined service keyword (such as **web-cache**). This information is used to validate that service group members are all using or providing the same service.

The content engines in a service group specify traffic to be redirected by protocol (TCP or UDP) and up to eight source or destination ports. Each service group has a priority status assigned to it. The priority of a dynamic service is assigned by the content engine. The priority value is in the range of 0 to 255 where 0 is the lowest priority. The predefined web-cache service has an assigned priority of 240.

WCCPv2 Support for Multiple Devices

WCCPv2 allows multiple devices to be attached to a cluster of cache engines. The use of multiple devices in a service group allows for redundancy, interface aggregation, and distribution of the redirection load. WCCPv2 supports up to 32 devices per service group. Each service group is established and maintained independently.

WCCPv2 MD5 Security

WCCPv2 provides optional authentication that enables you to control which switches and content engines become part of the service group using passwords and the Hashed Message Authentication Code—Message Digest (HMAC MD5) standard. Shared-secret MD5 one-time authentication (set using the **ip wccp password** *password* global configuration command) enables messages to be protected against interception, inspection, and replay.

WCCPv2 Web Cache Packet Return

If a content engine is unable to provide a requested object it has cached due to error or overload, the content engine will return the request to the device for onward transmission to the originally specified destination server. WCCPv2 provides a check on packets that determines which requests have been returned from the content engine unserviced. Using this information, the device can then forward the request to the originally targeted server (rather than attempting to resend the request to the content engine cluster). This process provides error handling transparency to clients.

Typical reasons why a content engine would reject packets and initiate the packet return feature include the following:

- Instances when the content engine is overloaded and has no room to service the packets
- Instances when the content engine is filtering for certain conditions that make caching packets counterproductive (for example, when IP authentication has been turned on)

WCCPv2 Load Distribution

WCCPv2 can be used to adjust the load being offered to individual content engines to provide an effective use of the available resources while helping to ensure high quality of service (QoS) to the clients. WCCPv2 allows the designated content engine to adjust the load on a particular content engine and balance the load across the content engines in a cluster. WCCPv2 uses three techniques to perform load distribution:

- Hot spot handling: Allows an individual hash bucket to be distributed across all the content engines. Prior to WCCPv2, information from one hash bucket could go to only one content engine.
- Load balancing: Allows the set of hash buckets assigned to a content engine to be adjusted so that the load can be shifted from an overwhelmed content engine to other members that have available capacity.
- Load shedding: Enables the switch to selectively redirect the load to avoid exceeding the capacity of a content engine.

The use of these hashing parameters prevents one content engine from being overloaded and reduces the potential for bottlenecking.

WCCP Bypass Packets

WCCP intercepts IP packets and redirects those packets to a destination other than the destination that is specified in the IP header. Typically the packets are redirected from a web server on the Internet to a web cache that is local to the destination.

Occasionally a web cache cannot manage the redirected packets appropriately and returns the packets unchanged to the originating device. These packets are called bypass packets and are returned to the originating device using Layer 2 forwarding without encapsulation (L2). The device decapsulates and forwards the packets normally. The VRF associated with the ingress interface (or the global table if there is no VRF associated) is used to route the packet to the destination.

WCCP Closed Services and Open Services

In applications where packets are intercepted and redirected by a Cisco switch or a router to external WCCP client devices, it may be necessary to block the packets for the application when a WCCP client device is not available. This blocking is achieved by configuring a WCCP closed service. When a WCCP service is configured as closed, the packets that fulfill the services, but do not have an active client device, are discarded.

By default, WCCP operates as an open service, wherein communication between clients and servers proceeds normally in the absence of an intermediary device.

The **ip wccp service-list** command can be used for both closed-mode and open-mode services. Use the **service-list** keyword and *service-access-list* argument to register an application protocol type or port number. Use the **mode** keyword to select an open or closed service.

WCCP Outbound ACL Check

When WCCP is enabled for redirection on an ingress interface, the packets are redirected by WCCP and instead egress on an interface other than the destination that is specified in the IP header. The packets are still subject to ACLs configured on the ingress interface. However, redirection can cause the packets to bypass the ACL configured on the original egress interface. Packets that would have been dropped because of the ACL configured on the original egress interface can be sent out on the redirect egress interface, which poses

a possible security problem. Enabling the WCCP Outbound ACL check feature ensures that redirected packets are subject to any ACL conditions configured on the original egress interface.

WCCP Service Groups

WCCP is a component of Cisco IOS XE software that redirects traffic with defined characteristics from its original destination to an alternative destination. The typical application of WCCP is to redirect traffic bound for a remote web server to a local web cache to improve response time and optimize network resource usage.

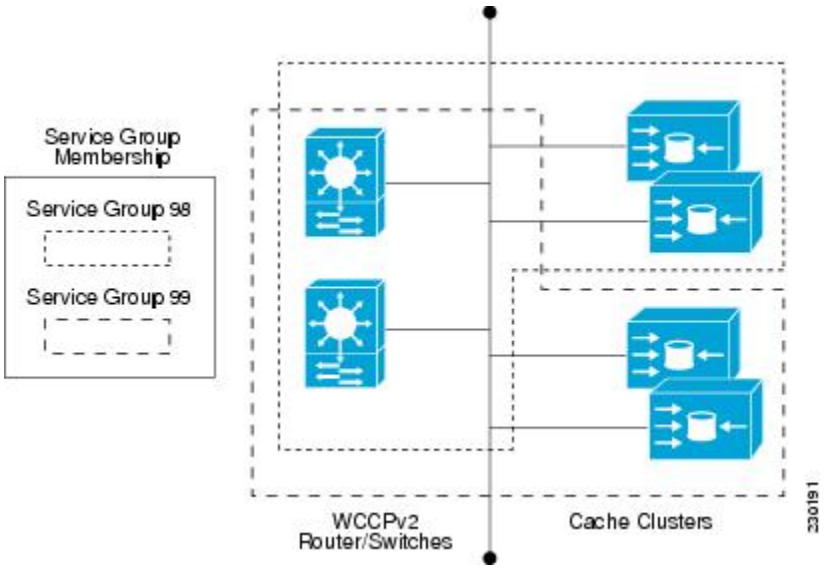
The nature of the selected traffic for redirection is defined by service groups (see figure below) specified on content engines and communicated to switches or routers using WCCP.

WCCPv2 supports up to 32 switches per service group. Each service group is established and maintained independently.

WCCPv2 uses service groups based on logical redirection services, deployed for intercepting and redirecting traffic. The standard service is web cache, which intercepts TCP port 80 (HTTP) traffic and redirects that traffic to the content engines. This service is referred to as a *well-known service*, because the characteristics of the web cache service are known by both the switch and content engines. A description of a well-known service is not required beyond a service identification. To specify the standard web cache service, use the **ip wccp** command with the **web-cache** keyword.

Note More than one service can run on a switch at the same time, and switches and content engines can be part of multiple service groups at the same time.

Figure 21: WCCP Service Groups



The dynamic services are defined by the content engines; the content engine instructs the switch which protocol or ports to intercept, and how to distribute the traffic. The switch itself does not have information on the characteristics of the dynamic service group's traffic, because this information is provided by the first content engine to join the group. In a dynamic service, up to eight ports can be specified within a single protocol.

Cisco Content Engines, for example, use dynamic service 99 to specify a reverse-proxy service. However, other content engine devices may use this service number for some other service.

WCCP: Check All Services

An interface may be configured with more than one WCCP service. When more than one WCCP service is configured on an interface, the precedence of a service depends on the relative priority of the service compared to the priority of the other configured services. Each WCCP service has a priority value as part of its definition. When an interface is configured with more than one WCCP service, the precedence of the packets is matched against service groups in priority order.

Note The priority of a WCCP service group cannot be configured through Cisco IOS XE software.

With the **ip wccp check services all** command, WCCP can be configured to check all configured services for a match and perform redirection for those services if appropriate. The caches to which packets are redirected can be controlled by a redirect ACL and by the service priority. The **ip wccp check services all** command must be configured at global level to support multiple WCCP services.

If no WCCP services are configured with a redirect ACL, the services are considered in priority order until a service is found that matches the IP packet. If no services match the packet, the packet is not redirected. If a service matches the packet and the service has a redirect ACL configured, then the IP packet will be checked against the ACL. If the packet is rejected by the ACL, the packet will not be passed down to lower priority services unless the **ip wccp check services all** command is configured. When the **ip wccp check services all** command is configured, WCCP will continue to attempt to match the packet against any remaining lower priority services configured on the interface.

WCCP VRF

WCCP VRF enhances the WCCPv2 protocol by adding support for virtual routing and forwarding (VRF). WCCP VRF allows service groups to be configured on a per-VRF basis in addition to being defined globally. Along with the service identifier, the VRF of the WCCP packets arriving at the device is used to associate the cache engine with a configured service group. The same WCCP VRF must have the interface on which redirection is applied, the interface that is connected to the cache engine, and the interface on which the packet would have left if it had not been redirected.

WCCP Troubleshooting Tips

CPU usage may be very high when WCCP is enabled. The WCCP counters enable a determination of the bypass traffic directly on the switch and can indicate whether the cause is high CPU usage due to enablement of WCCP. In some situations, 10 percent bypass traffic may be normal; in other situations, 10 percent may be high. However, any figure above 25 percent should prompt a closer investigation of what is occurring in the web cache.

If the counters suggest that the level of bypass traffic is high, the next step is to examine the bypass counters in the content engine and determine why the content engine is choosing to bypass the traffic. You can log in to the content engine console and use the CLI to investigate further. The counters allow you to determine the percent of traffic being bypassed.

You can use the **clear wccp** command to remove all WCCP statistics (counts) maintained on the device for a particular service.

You can use the **show wccp** command to display all WCCP global statistics (counts).

How to Configure WCCP

The following configuration tasks assume that you have already installed and configured the content engines you want to include in your network. You must configure the content engines in the cluster before configuring the WCCP functionality on your routers or switches. See Cisco Cache Engine User Guide for content engine configuration and setup tasks.

Configuring WCCP

Perform this task to configure the WCCP.

Until you configure a WCCP service using the **ip wccp**{**web-cache** | *service-number*} global configuration command, WCCP is disabled on a device.

Use the **ip wccp web-cache password** command to set a password for a device and the content engines in a service group. MD5 password security requires that each device and content engine that wants to join a service group be configured with the service group password. The password must be up to eight characters in length. Each content engine or device in the service group authenticates the security component in a received WCCP packet immediately after validating the WCCP message header. Packets failing authentication are discarded.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> 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 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} [**group-address** *multicast-address*] [**redirect-list** *access-list*] [**group-list** *access-list*] [**password** *password* [**0** | **7**]] **Example:** `Device(config)# ip wccp web-cache password pwd`	Specifies a web cache or dynamic service to enable on a device, specifies the IP multicast address used by the service group, specifies the access lists to use, specifies whether to use MD5 authentication, and enables the WCCP service. **Note** The password length must not exceed eight characters.

	Command or Action	Purpose
Step 4	**interface** *type number* **Example:** `Device(config)# interface Gigabitethernet 0/0`	Targets an interface number for which the web cache service will run, and enters interface configuration mode.
Step 5	**ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **redirect** {**in** | **out**} **Example:** `Device(config-if)# ip wccp web-cache redirect in`	Enables packet redirection on an outbound or inbound interface using WCCP. As indicated by the **out** and **in** keyword options, redirection can be specified for outbound interfaces or inbound interfaces.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 7	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/2/0`	Targets an interface number on which to exclude traffic for redirection, and enters interface configuration mode.
Step 8	**ip wccp redirect exclude in** **Example:** `Device(config-if)# ip wccp redirect exclude in`	(Optional) Excludes redirect traffic from the specified interface.

Configuring Closed Services

Perform this task to specify the number of service groups for WCCP, to configure a service group as a closed or open service, and to optionally specify a check of all services.

Procedure

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

	Command or Action	Purpose
	`Device# configure terminal`	

Step 3	Enter one of the following commands:
	• **ip wccp** [**vrf** *vrf-name*] *service-number* [**service-list** *service-access-list* **mode** {**open** | **closed**}]
	• **ip wccp** [**vrf** *vrf-name*] **web-cache mode** {**open** | **closed**}
	Example:
	`Device(config)# ip wccp 90 service-list 120 mode closed`
	or
	`Device(config)# ip wccp web-cache mode closed`
		Configures a dynamic WCCP service as closed or open.
		or
		Configures a web cache service as closed or open.
		Note When configuring the web cache service as a closed service, you cannot specify a service access list.
		Note When configuring a dynamic WCCP service as a closed service, you must specify a service access list.

Step 4	**ip wccp** [**vrf** *vrf-name*] **check services all** **Example:** `Device(config)# ip wccp check services all`	(Optional) Enables a check of all the WCCP services. Use this command to configure WCCP to check the other configured services for a match and perform redirection for those services if appropriate. The caches to which packets are redirected can be controlled by the redirect ACL and not just the service description. **Note** The **ip wccp check services all** command is a global WCCP command that applies to all services and not associated with just a single service.
Step 5	**ip wccp** {**web-cache** | *service-number*} **Example:** `Device(config)# ip wccp 201`	Specifies the WCCP service identifier. • You can specify the standard web cache service or a dynamic service number from 0 to 255. • The maximum number of services that can be specified is 256.
Step 6	**exit** **Example:** `Device(config)# exit`	Exits to privileged EXEC mode.

Registering a Device to a Multicast Address

If you decide to use the multicast address option for your service group, you must configure the device to listen for the multicast broadcasts on an interface.

For network configurations where redirected traffic needs to traverse an intervening device, the device being traversed must be configured to perform IP multicast routing. You must configure the following two components to enable traversal over an intervening device:

- Enable IP multicast routing using the **ip multicast-routing** global configuration command.
- Enable the interfaces to which the cache engines will connect to receive multicast transmissions using the **ip wccp group-listen** interface configuration command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip multicast-routing** [**vrf** *vrf-name*] [**distributed**] **Example:** `Device(config)# ip multicast-routing`	Enables IP multicast routing.
Step 4	**ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **group-address** *multicast-address* **Example:** `Device(config)# ip wccp 99 group-address 239.1.1.1`	Specifies the multicast address for the service group.
Step 5	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Enables the interfaces for which the web cache service will run, and enters interface configuration mode. These interfaces connect to the content engines to receive multicast transmissions.
Step 6	**ip pim** {**sparse-mode** | **sparse-dense-mode** | **dense-mode** [**proxy-register** {**list** *access-list* | **route-map** *map-name*}]}	(Optional) Enables Protocol Independent Multicast (PIM) on an interface.

	Command or Action	Purpose
	Example: `Device(config-if)# ip pim dense-mode`	**Note** To ensure correct operation of the **ip wccp group-listen** command on the Catalyst 9000 Series switches, enter the **ip pim** command in addition to the **ip wccp group-listen** command.
Step 7	**ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **group-listen** **Example:** `Device(config-if)# ip wccp 99 group-listen`	Configures an interface to enable or disable the reception of IP multicast packets for WCCP.

Using Access Lists for a WCCP Service Group

Perform this task to configure the device to use an access list to determine which traffic should be directed to which content engine.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *access-list-number* **remark** *remark* **Example:** `Device(config)# access-list 1 remark Give access to user1`	(Optional) Adds a user-friendly comment about an access list entry. A remark of up to 100 characters can precede or follow an access list entry.
Step 4	**access-list** *access-list-number* **permit** {*source* [*source-wildcard*] | **any**} [**log**] **Example:** `Device(config)# access-list 1 permit 172.16.5.22 0.0.0.0`	Creates an access list that enables or disables traffic redirection to the cache engine and permits the specified source based on a source address and wildcard mask. • Every access list needs at least one permit statement; it does not need to be the first entry.

	Command or Action	Purpose
		• Standard IP access lists are numbered 1 to 99 or 1300 to 1999. • If the *source-wildcard* is omitted, a wildcard mask of 0.0.0.0 is assumed, which means a match on all bits of the source address. • The keyword **any** can be used as a substitute for *source source-wildcard* to specify the source and source wildcard of 0.0.0.0 255.255.255.255. • In this example, host 172.16.5.22 is allowed to pass the access list.
Step 5	**access-list** *access-list-number* **remark** *remark* **Example:** `Device(config)# access-list 1 remark Give access to user1`	(Optional) Adds a user-friendly comment about an access list entry. A remark of up to 100 characters can precede or follow an access list entry.
Step 6	**access-list** *access-list-number* **deny** {*source* [*source-wildcard*] | **any**} | [**log**] **Example:** `Device(config)# access-list 1 deny 172.16.7.34 0.0.0.0`	Denies the specified source based on a source address and wildcard mask. • If the *source-wildcard* is omitted, a wildcard mask of 0.0.0.0 is assumed, which means a match on all bits of the source address. • The keyword **any** can be used as a substitute for the *source source-wildcard* to specify the source and source wildcard of 0.0.0.0 255.255.255.255. • In this example, host 172.16.7.34 cannot pass the access list.
Step 7	Repeat combinations of Step 3 through Step 6 until you have specified the sources on which you want to base your access list.	Remember that all the sources that are not specifically permitted are denied by an implicit **deny** statement at the end of the access list.
Step 8	**ip wccp** [**vrf** *vrf-name*] **web-cache group-list** *access-list* **Example:** `Device(config) ip wccp web-cache group-list 1`	Indicates to the device about which IP addresses of content engines should be accepted from.
Step 9	**ip wccp** [**vrf** *vrf-name*] **web-cache redirect-list** *access-list*	(Optional) Disables caching for certain clients.

	Command or Action	Purpose
	Example: `Device(config)# ip wccp web-cache redirect-list 1`	

Enabling the WCCP Outbound ACL Check

Note When all the redirection is performed in the hardware, the mode of redirection changes when outbound ACL checking is enabled. The first packet is switched in the software to allow an extra ACL check to be performed before a shortcut is installed.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> 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 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} [**group-address** *multicast-address*] [**redirect-list** *access-list*] [**group-list** *access-list*] [**password** *password*] **Example:** `Device(config)# ip wccp web-cache`	Enables support for a Cisco content engine service group, or any content engine service group and configures a redirect ACL list or group ACL. **Note** The **web-cache** keyword is for WCCP Version 1 and Version 2 and the *service-number* argument is for WCCP Version 2 only.
Step 4	**ip wccp check acl outbound** **Example:** `Device(config)# ip wccp check acl outbound`	Checks the access control list (ACL) for egress interfaces in the context of packets redirected by WCCP.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration.

Verifying and Monitoring WCCP Configuration Settings

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip wccp** [**web-cache** |*service-number*] [**detail view**] **Example:** `Device# show ip wccp 24 detail`	Displays global information related to WCCP, including the protocol version running, the number of content engines in the router service group, which content engine group is allowed to connect to the router, and which access list is being used. • *service-number*—(Optional) Dynamic number of the web-cache service group being controlled by the content engine. The range is from 0 to 99. For web caches that use Cisco Content Engines, the reverse proxy service is indicated by a value of 99. • **web-cache**—(Optional) statistics for the web-cache service. • **detail**—(Optional) other members of a particular service group or web cache that have or have not been detected. • **view**—(Optional) information about a router or all web caches.
Step 3	**show ip interface** **Example:** `Device# show ip interface`	Displays status about whether any **ip wccp redirection** commands are configured on an interface; for example, "Web Cache Redirect is enabled / disabled."
Step 4	**more system:running-config** **Example:** `Device# more system:running-config`	(Optional) Displays contents of the running configuration file (equivalent to the **show running-config** command).

Configuration Examples for WCCP

The following sections provide examples associated with WCCP configuration.

Example: Configuring a General WCCPv2 Session

```
Device# configure terminal
Device(config)# ip wccp web-cache group-address 224.1.1.100 password password
Device(config)# ip wccp source-interface GigabitEthernet 0/1/0
Device(config)# ip wccp check services all
! Configures a check of all WCCP services.
Device(config)# interface GigabitEthernet 0/1/0
Device(config-if)# ip wccp web-cache redirect in
Device(config-if)# exit
Device(config)# interface GigabitEthernet 0/2/0
Device(config-if)# ip wccp redirect exclude in
Device(config-if)# exit
```

Example: Setting a Password for a Device and Content Engines

```
Device# configure terminal
Device(config)# ip wccp web-cache password password1
```

Example: Configuring a Web Cache Service

```
Device# configure terminal
Device(config)# ip wccp web-cache
Device(config)# interface GigabitEthernet 0/1/0
Device(config-if)# ip wccp web-cache redirect in
Device(config-if)# exit
Device# copy running-config startup-config
```

The following example shows how to configure a session in which redirection of HTTP traffic arriving on Gigabit Ethernet interface 0/1/0 is enabled:

```
Device# configure terminal
Device(config)# interface GigabitEthernet 0/1/0
Device(config-if)# ip wccp web-cache redirect in
Device(config-if)# exit
Device# show ip interface GigabitEthernet 0/1/0
.
.
.
WCCP Redirect inbound is enabled
WCCP Redirect exclude is disabled
.
.
.
```

Example: Running a Reverse Proxy Service

The following example assumes that you are configuring a service group using Cisco cache engines, which use dynamic service 99 to run a reverse proxy service:

```
Device# configure terminal
Device(config)# ip wccp 99
```

```
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ip wccp 99 redirect out
```

Example: Registering a Device to a Multicast Address

```
Device# configure terminal
Device(config)# ip wccp web-cache group-address 224.1.1.100
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ip wccp web-cache group-listen
```

The following example shows a device configured to run a reverse proxy service, using the multicast address of 224.1.1.1. Redirection applies to packets going out through the Gigabit Ethernet interface 0/1/0:

```
Device# configure terminal
Device(config)# ip wccp 99 group-address 224.1.1.1
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ip wccp 99 redirect out
```

Example: Using Access Lists

To achieve better security, you can use a standard access list to notify the device which IP addresses are valid addresses for a content engine attempting to register with the current device. The following example shows a standard access list configuration session where the access list number is 10 for some sample hosts:

```
Device(config)# access-list 10 permit host 10.1.1.1
Device(config)# access-list 10 permit host 10.1.1.2
Device(config)# access-list 10 permit host 10.1.1.3
Device(config)# ip wccp web-cache group-list 10
```

To disable caching for certain clients, servers, or client/server pairs, you can use WCCP access lists. The following example shows that any requests coming from 10.1.1.1 to 10.3.1.1 will bypass the cache, and that all other requests will be serviced normally:

```
Device(config)# ip wccp web-cache redirect-list 120
Device(config)# access-list 120 deny tcp host 10.1.1.1 any
Device(config)# access-list 120 deny tcp any host 10.3.1.1
Device(config)# access-list 120 permit ip any any
```

The following example configures a device to redirect web-related packets received via Gigabit Ethernet interface 0/1/0, destined to any host except 209.165.200.224:

```
Device(config)# access-list 100 deny ip any host 209.165.200.224
Device(config)# access-list 100 permit ip any any
Device(config)# ip wccp web-cache redirect-list 100
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ip wccp web-cache redirect in
```

Example: WCCP Outbound ACL Check Configuration

The following configuration example shows that the access list prevents traffic from network 10.0.0.0 leaving Gigabit Ethernet interface 0/1/0. Because the outbound ACL check is enabled, WCCP does not redirect that traffic. WCCP checks packets against the ACL before they are redirected.

```
Device(config)# ip wccp web-cache
Device(config)# ip wccp check acl outbound
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ip access-group 10 out
Device(config-if)# exit
Device(config)# ip wccp web-cache redirect-list redirect-out
Device(config)# access-list 10 deny 10.0.0.0 0.255.255.255
Device(config)# access-list 10 permit any
```

If the outbound ACL check is disabled, the HTTP packets from network 10.0.0.0 would be redirected to a web cache. Users with that network address could retrieve web pages even though the network administrator wanted to prevent it.

Example: Verifying WCCP Settings

The following example shows how to verify your configuration changes by using the **more system:running-config** command in privileged EXEC mode. The following example shows that both the web cache service and dynamic service 99 are enabled on the device:

```
Device# more system:running-config

 Building configuration...
 Current configuration:
 !
 version 12.0
 service timestamps debug uptime
 service timestamps log uptime
 no service password-encryption
 service udp-small-servers
 service tcp-small-servers
 !
 hostname router4
 !
 enable secret 5 $1$nSVy$faliJsVQXVPW.KuCxZNTh1
 enable password password1
 !
 ip subnet-zero
 ip wccp web-cache
 ip wccp 99
 ip domain-name cisco.com
 ip name-server 10.1.1.1
 ip name-server 10.1.1.2
 ip name-server 10.1.1.3
 !
 !
 !
 interface GigabitEthernet0/1/1
 ip address 10.3.1.2 255.255.255.0
 no ip directed-broadcast
 ip wccp web-cache redirect in
 ip wccp 99 redirect in
 no ip route-cache
 no ip mroute-cache
 !
 interface GigabitEthernet0/1/0
 ip address 10.4.1.1 255.255.255.0
 no ip directed-broadcast
 ip wccp 99 redirect in
 no ip route-cache
 no ip mroute-cache
```

```
!
interface Serial0
no ip address
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
shutdown
!
interface Serial1
no ip address
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
shutdown
!
ip default-gateway 10.3.1.1
ip classless
ip route 0.0.0.0 0.0.0.0 10.3.1.1
no ip http server
!
!
!
line con 0
transport input none
line aux 0
transport input all
line vty 0 4
password password1
login
!
end
```

The following example shows how to display global statistics related to WCCP:

```
Device# show ip wccp web-cache detail

WCCP Client information:
WCCP Client ID:      10.1.1.2
Protocol Version:    2.0
State:               Usable
Redirection:         L2
Packet Return:       L2
Packets Redirected:  0
Connect Time:        00:20:34
Assignment:          MASK
Mask  SrcAddr    DstAddr    SrcPort DstPort
----  -------    -------    ------- -------
0000: 0x00000000 0x00001741 0x0000  0x0000
Value SrcAddr    DstAddr   SrcPort DstPort CE-IP
----- -------    -------   ------- ------- -----
0000: 0x00000000 0x00000000 0x0000 0x0000 0x3C010102 (10.1.1.2)
0001: 0x00000000 0x00000001 0x0000 0x0000 0x3C010102 (10.1.1.2)
0002: 0x00000000 0x00000040 0x0000 0x0000 0x3C010102 (10.1.1.2)
0003: 0x00000000 0x00000041 0x0000 0x0000 0x3C010102 (10.1.1.2)
0004: 0x00000000 0x00000100 0x0000 0x0000 0x3C010102 (10.1.1.2)
0005: 0x00000000 0x00000101 0x0000 0x0000 0x3C010102 (10.1.1.2)
0006: 0x00000000 0x00000140 0x0000 0x0000 0x3C010102 (10.1.1.2)
```

For more information about the **show ip wccp web-cache** command, see *Cisco IOS IP Application Services Command Reference*.

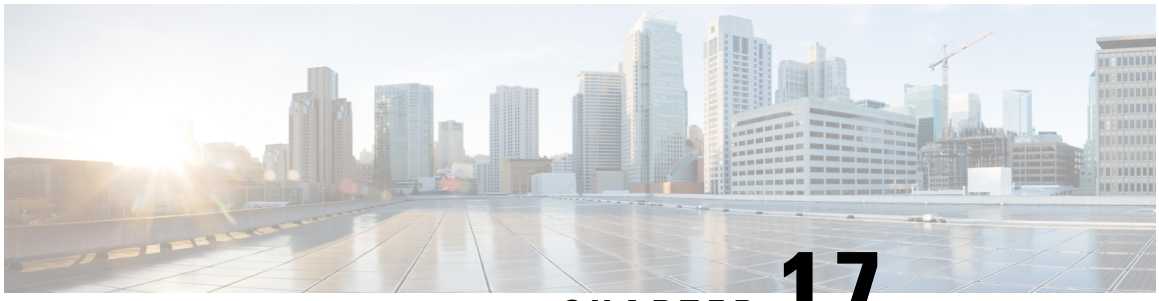
Feature History for WCCP

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	WCCP Support on Cisco Catalyst 9600 Series Swtiches	The Web Cache Communication Protocol (WCCP) is a Cisco-developed content-routing technology that intercepts IP packets and redirects those packets to a destination other than the one specified in the IP packet.
Cisco IOS XE Bengaluru 17.6.1	WCCP VRF	WCCP VRF enhances the WCCPv2 protocol by implementing support for virtual routing and forwarding.
Cisco IOS XE Cupertino 17.7.1	WCCP Support WCCP VRF	These features were implemented on supervisor modules C9400X-SUP-2 and C9400X-SUP-2XL, which were introduced in this release.

Use Cisco Feature Navigator to find information about platform and software image support.



CHAPTER 17

Configuring Enhanced Object Tracking

- Restrictions for Enhanced Object Tracking, on page 221
- Information About Enhanced Object Tracking, on page 221
- How to Configure Enhanced Object Tracking, on page 223
- Monitoring Enhanced Object Tracking, on page 235
- Feature History for Enhanced Object Tracking, on page 235

Restrictions for Enhanced Object Tracking

All tracking configurations must be reconfigured on all Layer 3 subinterfaces after a device reloads. All reload operations on the device must be allowed to complete, so that all Layer 3 subinterfaces are active, before the tracking configuration are reconfigured.

Information About Enhanced Object Tracking

The following sections provide information about enhanced object tracking.

Enhanced Object Tracking Overview

Before the introduction of the Enhanced Object Tracking feature, Hot Standby Router Protocol (HSRP) had a simple tracking mechanism that allowed you to track the interface line-protocol state only. If the line-protocol state of the interface went down, the HSRP priority of the router was reduced, allowing another HSRP router with a higher priority to become active.

The Enhanced Object Tracking feature separates the tracking mechanism from HSRP and creates a separate standalone tracking process that can be used by processes other than HSRP. This feature allows the tracking of other objects in addition to the interface line-protocol state.

A client process such as HSRP, Virtual Router Redundancy Protocol (VRRP), or Gateway Load Balancing Protocol (GLBP), can register its interest in tracking objects and then be notified when the tracked object changes state.

Each tracked object has a unique number that is specified in the tracking command-line interface (CLI). Client processes use this number to track a specific object. The tracking process periodically polls the tracked object for value changes and sends any changes (as up or down values) to interested client processes, either

immediately or after a specified delay. Several clients can track the same object, and can take different actions when the object changes state.

You can also track a combination of objects in a list by using either a weight threshold or a percentage threshold to measure the state of the list. You can combine objects using Boolean logic. A tracked list with a Boolean "AND" function requires that each object in the list be in an up state for the tracked object to be up. A tracked list with a Boolean "OR" function needs only one object in the list to be in the up state for the tracked object to be up.

Tracking Interface Line-Protocol or IP Routing State

You can track either the interface line protocol state or the interface IP routing state. When you track the IP routing state, these three conditions are required for the object to be up:

- IP routing must be enabled and active on the interface.
- The interface line-protocol state must be up.
- The interface IP address must be known.

If all three of these conditions are not met, the IP routing state is down.

Tracked Lists

You can configure a tracked list of objects with a Boolean expression, a weight threshold, or a percentage threshold. A tracked list contains one or more objects. An object must exist before it can be added to the tracked list.

- You configure a Boolean expression to specify calculation by using either "AND" or "OR" operators.
- When you measure the tracked list state by a weight threshold, you assign a weight number to each object in the tracked list. The state of the tracked list is determined by whether or not the threshold was met. The state of each object is determined by comparing the total weight of all objects against a threshold weight for each object.
- When you measure the tracked list by a percentage threshold, you assign a percentage threshold to all objects in the tracked list. The state of each object is determined by comparing the assigned percentages of each object to the list.

Tracking Other Characteristics

You can also use the enhanced object tracking for tracking other characteristics.

- You can track the reachability of an IP route by using the **track ip route reachability** global configuration command.
- You can use the **track ip route metric threshold** global configuration command to determine if a route is above or below threshold.
- You can use the **track resolution** global configuration command to change the metric resolution default values for routing protocols.
- You can use the **track timer tracking** configuration command to configure the tracking process to periodically poll tracked objects.

Use the **show track** privileged EXEC command to verify enhanced object tracking configuration.

IP SLAs Object Tracking

Cisco IOS IP Service Level Agreements (IP SLAs) is a network performance measurement and diagnostics tool that uses active monitoring by generating traffic to measure network performance. Cisco IP SLAs operations collects real-time metrics that you can use for network troubleshooting, design, and analysis.

Object tracking of IP SLAs operations allows clients to track the output from IP SLAs objects and use this information to trigger an action. Every IP SLAs operation maintains an SNMP operation return-code value, such as OK or OverThreshold, that can be interpreted by the tracking process. You can track two aspects of IP SLAs operation: state and reachability. For state, if the return code is OK, the track state is up; if the return code is not OK, the track state is down. For reachability, if the return code is OK or OverThreshold, reachability is up; if not OK, reachability is down.

Static Route Object Tracking

Static routing support using enhanced object tracking provides the ability for the device to use ICMP pings to identify when a pre-configured static route or a DHCP route goes down. When tracking is enabled, the system tracks the state of the route and informs the client when that state changes. Static route object tracking uses Cisco IP SLAs to generate ICMP pings to monitor the state of the connection to the primary gateway.

How to Configure Enhanced Object Tracking

The following sections provide configuration information about enhanced object tracking.

Configuring Tracking for Line State Protocol or IP Routing State on an Interface

Follow these steps to track the line-protocol state or IP routing state of an interface:

Procedure

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

	Command or Action	Purpose
Step 3	**track** *object-number* **interface** *interface-id* **line-protocol** **Example:** `Device(config)# track 33 interface gigabitethernet 1/0/1 line-protocol`	(Optional) Creates a tracking list to track the line-protocol state of an interface and enter tracking configuration mode. • The object-number identifies the tracked object and can be from 1 to 500. • The**interface** interface-id is the interface being tracked.
Step 4	**delay** { *object-number* **up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time in seconds to delay communicating state changes of a tracked object. The range is from 1 to 180 seconds.
Step 5	**exit**	Returns to global configuration mode.
Step 6	**track** *object-number* **interface** *interface-id* **ip routing** **Example:** `Device(config)# track 33 interface gigabitethernet 1/0/1 ip routing`	(Optional) Creates a tracking list to track the IP routing state of an interface and enter tracking configuration mode. IP route tracking tracks an IP route in the routing table and the ability of an interface to route IP packets. • The object-number identifies the tracked object and can be from 1 to 500. • The**interface** interface-id is the interface being tracked.
Step 7	**delay** { *object-number* **up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time in seconds to delay communicating state changes of a tracked object. The range is from 1 to 180 seconds.
Step 8	**end**	Returns to privileged EXEC mode.
Step 9	**show track***object-number*	Verifies that the specified objects are being tracked.

Configuring Tracked Lists

The following sections provide configuration information about tracked lists.

Configuring a Tracked List with a Weight Threshold

To track by weight threshold, configure a tracked list of objects, specify that weight is used as the threshold, and configure a weight for each of its objects. The state of each object is determined by comparing the total weight of all objects that are up against a threshold weight for each object.

You cannot use the Boolean “NOT” operator in a weight threshold list.

Follow these steps to configure a tracked list of objects by using a weight threshold and to configure a weight for each object:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *track-number* **list threshold** {**weight**} **Example:** `Device(config)# track 4 list threshold weight`	Configures a tracked list object, and enters tracking configuration mode. The track-number can be from 1 to 500. • **threshold—**Specifies the state of the tracked list based on a threshold. • **weight—** Specifies that the threshold is based on weight.
Step 4	**object** *object-number* [**weight** *weight-number*] **Example:** `Device(config)#object 2 weight 15`	Specifies the object to be tracked. The range is from 1 to 500. The optional**weight***weight-number* specifies the threshold weight for the object. The range is from 1 to 255. **Note** An object must exist before you can add it to a tracked list.
Step 5	**threshold weight** {**up** *number* | [**down** *number*]} **Example:** `Device(config-track)# threshold weight up 30 down 10`	(Optional) Specifies the threshold weight. • **up***number*—The range is from 1 to 255. • **down***number*—(Optional)The range depends on the number selected for the **up***number*. If you configure the **up***number* as 25, the range shown for the down number is 0 to 24.
Step 6	**delay** { **up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time in seconds to delay communicating state changes of a

	Command or Action	Purpose
		tracked object. The range is from 1 to 180 seconds.
Step 7	**end**	Returns to privileged EXEC mode.
Step 8	**show track** *object-number*	Verify that the specified objects are being tracked.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Tracked List with a Percentage Threshold

To track by percentage threshold, configure a tracked list of objects, specify that a percentage will be used as the threshold, and specify a percentage for all objects in the list. The state of the list is determined by comparing the assigned percentage of each object to the list.

You cannot use the Boolean "NOT" operator in a percentage threshold list.

Follow these steps to configure a tracked list of objects by using a percentage threshold:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *track-number* **list threshold** {**percentage**} **Example:** `Device(config)# track 4 list threshold percentage`	Configures a tracked list object, and enters tracking configuration mode. The track-number can be from 1 to 500. • **threshold—**Specifies the state of the tracked list based on a threshold. • **percentage—** Specifies that the threshold is based on percentage.

	Command or Action	Purpose
Step 4	**object** *object-number* **Example:** `Device(config)# object 1`	Specifies the object to be tracked. The range is from 1 to 500. **Note** An object must exist before you can add it to a tracked list.
Step 5	**threshold percentage** {**up** *number* | [**down***number*]} **Example:** `Device(config)# threshold percentage up 51 down 10`	(Optional) Specifies the threshold percentage. • **up***number*—The range is from 1 to 100. • **down***number*—(Optional)The range depends on the number selected for the **up***number*. If you configure the **up***number* as 25, the range shown for the down number is 0 to 24.
Step 6	**delay** { **up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time in seconds to delay communicating state changes of a tracked object. The range is from 1 to 180 seconds.
Step 7	**end**	Returns to privileged EXEC mode.
Step 8	**show track** *object-number*	Verify that the specified objects are being tracked.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring HSRP Object Tracking

Follow these steps to configure a standby HSRP group to track an object and change the HSRP priority based on the object state:

Procedure

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *object-number* {**interface** *interface-id* {**line-protocol** | **ip routing**} | **ip route** *ip address/prefix-length* {**metric threshold** | **reachability**} **list** {**boolean** {**and** | **or**}} | {**threshold** {**weight** | **percentage**}}}	(Optional) Create a tracking list to track the configured state and enter tracking configuration mode. • The object-number identifies the tracked object and can be from 1 to 500. • Enter**interface** *interface-id* to select an interface to track. • Enter**line-protocol** to track the interface line protocol state or enter **ip routing** to track the interface IP routing state . • Enter**ip route** *ip-address/prefix-length* to track the state of an IP route. • Enter**metric threshold** to track the threshold metric or enter **reachability** to track if the route is reachable. The default up threshold is 254 and the default down threshold is 255. • Enter **list** to track objects grouped in a list. **Note** Repeat this step for each interface to be tracked.
Step 4	**exit**	Return to global configuration mode.
Step 5	**interface** { *interface-id*	Enter interface configuration mode.
Step 6	**standby** [*group-number*] **ip** [*ip-address* **secondary**]]	Creates (or enables) the HSRP group by using its number and virtual IP address. • (Optional) *group-number*—Enters a group number on the interface for which HSRP is being enabled. The range is 0 to 255; the default is 0. If there is only one HSRP group, you do not need to enter a group number. • (Optional on all but one interface) *ip-address*—Specifies the virtual IP

	Command or Action	Purpose
		address of the hot standby router interface. You must enter the virtual IP address for at least one of the interfaces; it can be learned on the other interfaces. • (Optional) **secondary**—Specifies that the IP address is a secondary hot standby router interface. If this keyword is omitted, the configured address is the primary IP address.
Step 7	**standby** [*group-number*] **track** [*object-number* [**decrement** *priority-decrement*]]	Configures HSRP to track an object and change the hot standby priority based on the state of the object. • (Optional) *group-number*—Enters the group number to which the tracking applies. • *object-number*—Enters a number representing the object to be tracked. The range is from 1 to 500; the default is 1. • (Optional) **secondary**—Specifies that the IP address is a secondary hot standby router interface. If this keyword is omitted, the configured address is the primary IP address. • (Optional) **decrement** *priority-decrement*—Specifies the amount by which the hot standby priority for the router is decremented (or incremented) when the tracked object goes down (or comes back up). The range is from 1 to 255; the default is 10.
Step 8	**end**	Returns to privileged EXEC mode.
Step 9	**show standby**	Verifies the standby router IP address and tracking states.
Step 10	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring IP SLAs Object Tracking

Follow these steps to track the state of an IP SLAs operation or the reachability of an IP SLAs IP host:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *object-number* **ip sla** *operation-number* {**state** | **reachability**} **Example:** `Device(config)# track 2 ip sla 123 state`	Enters tracking configuration mode to track the state of an IP SLAs operation. • *object-number* range is from 1 to 500. • *operation-number* range is from 1 to 2147483647.
Step 4	**delay** { **up***seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time in seconds to delay communicating state changes of a tracked object. The range is from 1 to 180 seconds.
Step 5	**end**	Returns to privileged EXEC mode.
Step 6	**show track** *object-number*	Verifies that the specified objects are being tracked.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Static Route Object Tracking

The following sections provide configuration information about static route object tracking.

Configuring a Primary Interface for Static Routing

Follow these steps to configure a primary interface for static routing:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id*	Selects a primary or secondary interface and enters interface configuration mode.
Step 4	**description** *string*	Adds a description to the interface.
Step 5	**ip address** *ip-address mask* [**secondary**]	Sets the primary or secondary IP address for the interface.
Step 6	**exit**	Returns to global configuration mode.

Configuring a Primary Interface for DHCP

Follow these steps to configure a primary interface for DHCP:

Procedure

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

	Command or Action	Purpose
Step 3	**interface** *interface-id*	Selects a primary or secondary interface and enters interface configuration mode.
Step 4	**description** *string*	Adds a description to the interface.
Step 5	**ip dhcp client route track** *number*	Configures the DCHP client to associate any added routes with the specified track number. Valid numbers are from 1 to 500.
Step 6	**exit**	Returns to global configuration mode.

Configuring IP SLAs Monitoring Agent

You can configure an IP SLAs agent to ping an IP address using a primary interface and a track object to monitor the state of the agent.

Follow these steps to configure network monitoring with Cisco IP SLAs:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip sla** *operation number*	Begins configuring a Cisco IP SLAs operation and enters IP SLA configuration mode.
Step 4	**icmp-echo** { *destination ip-address* | *destination hostname* [**source - ipaddr** {*ip-address* | *hostname* **source-interface** *interface-id*]	Configures a Cisco IP SLAs end-to-end ICMP echo response time operation and enter IP SLAs ICMP echo configuration mode.
Step 5	**timeout** *milliseconds*	Sets the amount of time for which the operation waits for a response from its request packet.
Step 6	**frequency** *seconds*	Sets the rate at which the operation is sent into the network.

	Command or Action	Purpose
Step 7	**threshold** *milliseconds*	Sets the rising threshold (hysteresis) that generates a reaction event and stores history information for the operation.
Step 8	**exit**	Exits IP SLAs ICMP echo configuration mode.
Step 9	**ip sla schedule** *operation-number* [**life** {**forever** | *seconds*}] **start-time** *time* | **pending** | **now** | **after** *time*] **ageout** *seconds*] [**recurring**] **Example:** `Device(config)# track 2 200 state`	Configures the scheduling parameters for a single IP SLAs operation. • *object-number* range is from 1 to 500. • *operation-number* range is from 1 to 2147483647.
Step 10	**track** *object-number* **rtr** *operation-number* **state reachability**	Tracks the state of a Cisco IOS IP SLAs operation and enter tracking configuration mode.
Step 11	**end**	Returns to privileged EXEC mode.
Step 12	**show track** *object-number*	Verifies that the specified objects are being tracked.
Step 13	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring a Routing Policy and a Default Route

Follow these steps to configure a routing policy for backup static routing by using object tracking.

Procedure

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

	Command or Action	Purpose
Step 3	**access-list** *access-list-number*	Defines an extended IP access list. Configure any optional characteristics.
Step 4	**route-map** *map tag* [**permit** | **deny**] [*sequence-number*]	Enters route-map configuration mode and define conditions for redistributing routes from one routing protocol to another.
Step 5	**match ip address** {*access-list number* [**permit**|**deny**] [*sequence-number*]	Distribute any routes that have a destination network number address that is permitted by a standard or extended access list or performs policy routing on packets. You can enter multiple numbers or names.
Step 6	**set ip next-hop dynamic dhcp**	For DHCP networks only. Sets the next hop to the gateway that was most recently learned by the DHCP client.
Step 7	**set interface** *interface-id*	For static routing networks only. Indicates where to send output packets that pass a match clause of a route map for policy routing.
Step 8	**exit**	Returns to global configuration mode.
Step 9	**ip local policy route-map** *map tag*	Identifies a route map to use for local policy routing.
Step 10	**ip route** *prefix mask* {*ip address* | *interface-id* [*ip address*]} [*distance*] [*name*] [**permanent** | **track** *track-number*] [*tag tag*]	For static routing networks only. Establishes static routes. Entering **track***track-number* specifies that the static route is installed only if the configured track object is up.
Step 11	**end**	Returns to privileged EXEC mode.
Step 12	**show ip route track table**	Displays information about the IP route track table.
Step 13	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring Enhanced Object Tracking

Use the privileged EXEC or user EXEC commands in the table below, to display enhanced object tracking information.

Table 6: Commands for Displaying Tracking Information

Command	Purpose
show ip route track table	Displays information about the IP rou
show track [*object-number*]	Displays information about the all trac
show track brief	Displays VTP status and configuratio
show track interface [brief]	Displays information about tracked in
show track ip [*object-number*] [**brief**] **route**	Displays information about tracked IP
show track resolution	Displays the resolution of tracked para
show track timer	Displays tracked polling interval time

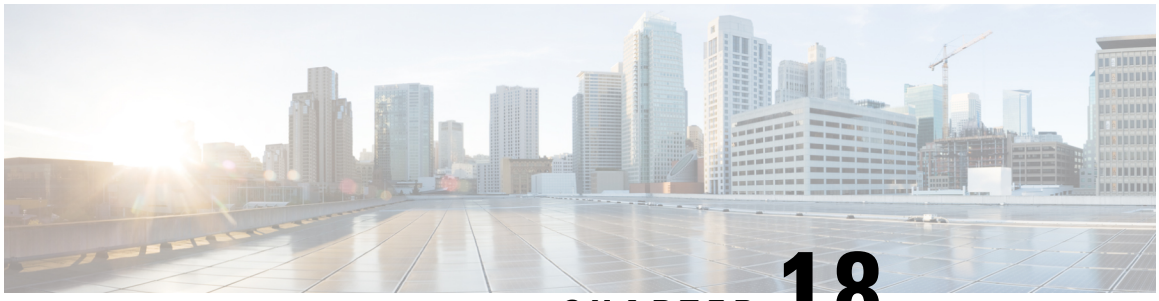
Feature History for Enhanced Object Tracking

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 Gibraltar 16.11.1	Enhanced Object Tracking	Enhanced object tracking allows advanced tracking compared to HSRP that allows interface line-protocol state tracking only.
Cisco IOS XE Cupertino 17.7.1	Enhanced Object Tracking	This feature was implemented on the Cisco Catalyst 9600 Series Supervisor 2 Module (C9600X-SUP-2).

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/



CHAPTER 18

Configuring TCP MSS Adjustment

- Restrictions for TCP MSS Adjustment, on page 237
- Information about TCP MSS Adjustment, on page 237
- How to Configure TCP MSS Adjustment, on page 238
- Configuration Examples for TCP MSS Adjustment, on page 239
- Feature History for TCP MSS Adjustment, on page 240

Restrictions for TCP MSS Adjustment

- Subinterfaces do not support TCP MSS Adjust.
- TCP MSS adjustment configuration works only if applied on an ingress interface. This configuration does not work if applied on an egress interface.

Information about TCP MSS Adjustment

The Transmission Control Protocol (TCP) Maximum Segment Size (MSS) Adjustment feature enables the configuration of the maximum segment size for transient packets that traverse a router, specifically TCP segments with the SYN bit set. Use the **ip tcp adjust-mss** command in interface configuration mode to specify the MSS value on the intermediate router of the SYN packets to avoid truncation.

When a host (usually a PC) initiates a TCP session with a server, it negotiates the IP segment size by using the MSS option field in the TCP SYN packet. The value of the MSS field is determined by the MTU configuration on the host. The default MSS value for a PC is 1500 bytes.

The PPP over Ethernet (PPPoE) standard supports an MTU of only 1492 bytes. The disparity between the host and PPPoE MTU size can cause the router in between the host and the server to drop 1500-byte packets and terminate TCP sessions over the PPPoE network. Even if the path MTU (which detects the correct MTU across the path) is enabled on the host, sessions may be dropped because system administrators sometimes disable the ICMP error messages that must be relayed from the host in order for path MTU to work.

The **ip tcp adjust-mss** command helps prevent TCP sessions from being dropped by adjusting the MSS value of the TCP SYN packets.

The **ip tcp adjust-mss** command is effective only for TCP connections passing through the router.

In most cases, the optimum value for the *max-segment-size* argument of the **ip tcp adjust-mss** command is 1452 bytes. This value plus the 20-byte IP header, the 20-byte TCP header, and the 8-byte PPPoE header add up to a 1500-byte packet that matches the MTU size for the Ethernet link.

Note TCP MSS adjustment-based traffic is always software switched.

Supported Interfaces

TCP MSS Adjust is supported only on the following interfaces:

- Physical Layer 3 interface
- SVI
- Layer 3 port channel
- Layer 3 GRE tunnel

How to Configure TCP MSS Adjustment

The following sections provide configuration information for TCP MSS adjustment.

Configuring the MSS Value for Transient TCP SYN Packets

Before you begin

Perform this task to configure the MSS for transient packets that traverse a router, specifically TCP segments with the SYN bit set.

We recommend that you use **ip tcp adjust-mss 1452** command.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted
Step 2	**configure terminal** **Example:** `Device#config terminal`	Enters the global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)#interface GigabitEthernet 1/0/0`	Configures an interface type and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ip tcp adjust-mss** *max-segment-size* **Example:** `Device(config-if)#ip tcp adjust-mss 1452`	Adjusts the MSS value of TCP SYN packets going through a router. The max-segment-size argument is the maximum segment size, in bytes. The range is from 500 to 1460.
Step 5	**end** **Example:** `Device(config-if)#end`	Exits to global configuration mode.

Configuring the MSS Value for IPv6 Traffic

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted
Step 2	**configure terminal** **Example:** `Device#config terminal`	Enters the global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)#interface GigabitEthernet 1/0/0`	Configures an interface type and enters interface configuration mode.
Step 4	**ipv6 tcp adjust-mss** *max-segment-size* **Example:** `Device(config-if)#ipv6 tcp adjust-mss 1440`	Adjusts the MSS value of TCP DF packets going through a device. The max-segment-size argument is the maximum segment size, in bytes. The range is from 40 to 1440.
Step 5	**end** **Example:** `Device(config-if)#end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for TCP MSS Adjustment

The following sections provide configuration examples for TCP MSS adjustment.

Example: Configuring the TCP MSS Adjustment

```
Device(config)#vpdn enable
Device(config)#no vpdn logging
Device(config)#vpdn-group 1
Device(config-vpdn)#request-dialin
Device(config-vpdn-req-in)#protocol pppoe
Device(config-vpdn-req-in)#exit
Device(config-vpdn)#exit
Device(config)#interface GigabitEthernet 0/0/0
Device(config-if)#ip address 192.168.100.1.255.255.255.0
Device(config-if)#ip tcp adjust-mss 1452
Device(config-if)#ip nat inside
Device(config-if)#exit
```

Example: Configuring the TCP MSS Adjustment for IPv6 traffic

```
Device>enable
Device#configure terminal
Device(config)#interface GigabitEthernet 0/0/0
Device(config)#ipv6 tcp adjust-mss 1440
Device(config)#end
```

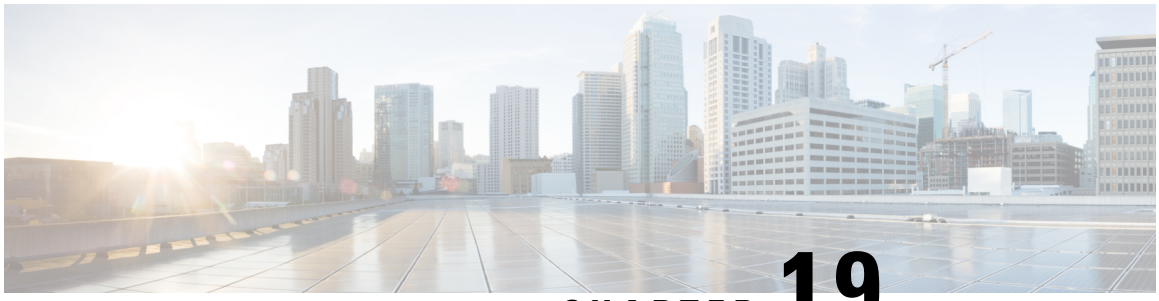
Feature History for TCP MSS Adjustment

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 Gibraltar 16.11.1	Transmission Control Protocol (TCP) Maximum Segment Size (MSS) Adjustment	The TCP MSS Adjustment feature enables the configuration of the maximum segment size for transient packets that traverse a router, specifically TCP segments with the SYN bitset. This feature helps prevent TCP sessions from being dropped by adjusting the MSS value of the TCP SYN packets.

Use the Cisco Feature Navigator to find information about platform and software image support.



CHAPTER 19

Enhanced IPv6 Neighbor Discovery Cache Management

- Enhanced IPv6 Neighbor Discovery Cache Management , on page 241
- Customizing the Parameters for IPv6 Neighbor Discovery , on page 242
- Examples: Customizing Parameters for IPv6 Neighbor Discovery, on page 243
- Additional References, on page 243
- Feature History for IPv6 Neighbor Discovery, on page 243

Enhanced IPv6 Neighbor Discovery Cache Management

Neighbor discovery protocol enforces the neighbor unreachability detection process to detect failing nodes, or devices, and the changes to link-layer addresses. Neighbor unreachability detection process maintains the reachability information for all the paths between hosts and neighboring nodes, including host-to-host, host-to-device, and device-to-host communication.

The neighbor cache maintains mapping information about the IPv6 link-local or global address to the link-layer address. The neighbor cache also maintains the reachability state of the neighbor using the neighbor unreachability detection process. Neighbors can be in one of the following five possible states:

- DELAY: Neighbor resolution is pending, and traffic might flow to this neighbor.
- INCOMPLETE: Address resolution is in progress, and the link-layer address is not yet known.
- PROBE: Neighbor resolution is in progress, and traffic might flow to this neighbor.
- REACHABLE: Neighbor is known to be reachable within the last reachable time interval.
- STALE: Neighbor requires resolution, and traffic may flow to this neighbor.

Use the **ipv6 nd na glean** command to configure the neighbor discovery protocol to glean an entry from an unsolicited neighbor advertisement.

Use the **ipv6 nd nud retry** command to configure the neighbor discovery protocol to maintain a neighbor discovery cache entry for a neighbor during a network disruption.

Use the **ipv6 nd cache expire refresh** command to configure the neighbor discovery protocol to maintain a neighbor discovery cache entry even when no traffic flows to the neighbor.

Customizing the Parameters for IPv6 Neighbor Discovery

To customize the parameters for IPv6 neighbor discovery, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1/4`	Specifies an interface type and identifier. Enters the interface configuration mode.
Step 4	**ipv6 nd nud retry** *base interval max-attempts* [*final-wait-time*] **Example:** `Device(config-if)# ipv6 nd nud retry 1 1000 3`	Configures the number of times neighbor unreachability detection resends neighbor solicitations.
Step 5	**ipv6 nd cache expire** *expire-time-in-seconds* [**refresh**] **Example:** `Device(config-if)# ipv6 nd cache expire 7200`	Configures the length of time before an IPv6 neighbor discovery cache entry expires.
Step 6	**ipv6 nd na glean** **Example:** `Device(config-if)# ipv6 nd na glean`	Configures the length of time before an IPv6 neighbor discovery cache entry expires.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 8	**show ipv6 interface** **Example:** `Device# show ipv6 interface`	(Optional) Displays the usability status of interfaces that are configured for IPv6 along with neighbor discovery cache management.

Examples: Customizing Parameters for IPv6 Neighbor Discovery

The following example shows that IPv6 neighbor advertisement gleaning is enabled and the IPv6 neighbor discovery cache expiry is set to 7200 seconds (2 hours):

```
Device> enable
Device# configure terminal
Device(config)# interface Port-channel 189
Device(config-if)# no ip address
Device(config-if)# ipv6 address 2001:BD8::/64
Device(config-if)# ipv6 nd reachable-time 2700000
Device(config-if)# ipv6 nd na glean
Device(config-if)# ipv6 nd cache expire 7200
Device(config-if)# no ipv6 redirects
Device(config-if)# end
```

Additional References

Related Documents

Related Topic	Document Title
For complete syntax and usage information for the commands used in this chapter.	See the *IP Addressing Services* section of *Command Reference (Catalyst 9600 Series Switches)*
For information on IPv6 Neighbor Discovery Inspection	See the *Security* section of *Software Configuration Guide (Catalyst 9600 Switches)*

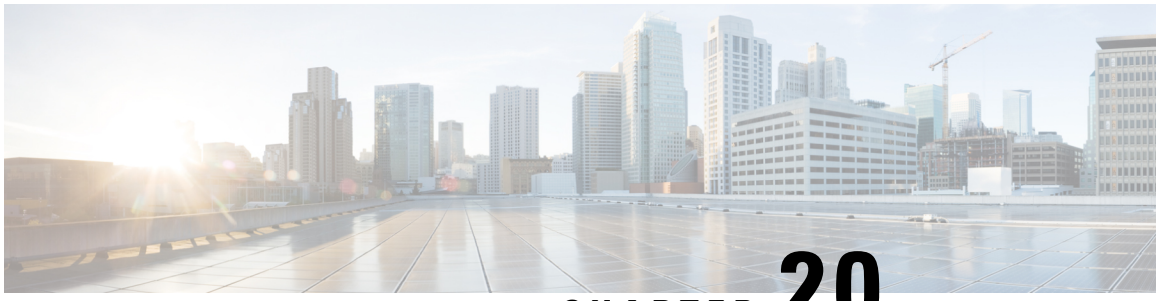
Feature History for IPv6 Neighbor Discovery

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 Gibraltar 16.11.1	Enhanced IPv6 Neighbor Discovery Cache Management	Neighbor discovery protocol enforces neighbor unreachability detection, which can detect failing nodes or routers, and changes to link-layer addresses.
Cisco IOS XE Cupertino 17.7.1	Enhanced IPv6 Neighbor Discovery Cache Management	This feature was implemented on the Cisco Catalyst 9600 Series Supervisor 2 Module (C9600X-SUP-2).

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/



CHAPTER 20

IPv6 Neighbor Discovery Proxy

- Prerequisites for IPv6 Neighbor Discovery Proxy, on page 245
- Restrictions for IPv6 Neighbor Discovery Proxy, on page 245
- Information About IPv6 Neighbor Discovery Proxy, on page 245
- How to Configure IPv6 Neighbor Discovery Proxy, on page 246
- Verifying IPv6 Neighbor Discovery Proxy, on page 251
- Configuration Examples For IPv6 Neighbor Discovery Proxy, on page 251
- Feature History for IPv6 Neighbor Discovery Proxy, on page 252

Prerequisites for IPv6 Neighbor Discovery Proxy

The following prerequisites are applicable when configuring IPv6 neighbor discovery proxy:

- Ensure that IPv6 is enabled on the Switch Virtual Interface (SVI).
- When you configure Duplicate Address Detection (DAD) proxy, ensure that device tracking is configured on the device.

Restrictions for IPv6 Neighbor Discovery Proxy

- IPv6 routing proxy is not supported on layer 3 interfaces.
- The IPv6 DAD proxy and routing proxy features are not supported on etherchannel ports.

Information About IPv6 Neighbor Discovery Proxy

IPv6 neighbor discovery proxy restricts IPv6 hosts within a VLAN from communicating directly with each other and allows them to communicate only via the gateway. A device operating as an IPv6 neighbor discovery proxy responds to packets on behalf of the target.

IPv6 neighbor discovery proxy operations are achieved using the following implementations:

IPv6 Routing-Proxy

A device operating as an IPv6 routing proxy listens to all neighbor discovery proxy messages sent on the link and responds unconditionally to neighbor solicitation lookup and neighbor-unreachability-detection messages with neighbor advertisement (setting the SVI MAC address in the TLLA option) on behalf of the destination hosts to attract the traffic to itself.

IPv6 DAD Proxy

IPv6 DAD proxy feature responds to DAD queries on behalf of a node that owns the queried address. IPv6 DAD proxy depends on a device tracking database to ensure uniqueness of IPv6 addresses.

When receiving a DAD request from a host for a target, the DAD proxy performs a lookup into the binding table, and if the lookup returns a location, it sends an neighbor solicitation neighbor-unreachability-detection message to verify that the target is still alive.

- If the target replies to the neighbor-unreachability-detection message, the DAD proxy sends back an neighbor advertisement to the host (setting the SVI MAC address in the TLLA option).
- If the device does not respond to the neighbor-unreachability-detection message, the DAD proxy does not send any response to DAD request.

How to Configure IPv6 Neighbor Discovery Proxy

•

Configuring IPv6 Routing Proxy in VLAN Configuration Mode

Before you begin

Follow these steps to enable IPv6 on an SVI:

```
Device# enable
Device# configure terminal
Device(config)# interface vlan vlan-id
Device(config-if)# no ipv6 redirects
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 address ipv6-address
```

To configure IPv6 routing proxy in VLAN configuration mode, follow this procedure:

Procedure

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

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**vlan configuration** *vlan-id* **Example:** `Device(config)# vlan configuration 15`	Enters the VLAN configuration mode. This mode allows you to name, set the state, disable, and shut down the VLAN or range of VLANs.
Step 4	**[no] ipv6 nd routing-proxy** **Example:** `Device(config-vlan)# ipv6 nd routing-proxy`	Specifies if the neighbor discovery suppress must operate in routing proxy mode.
Step 5	**end** **Example:** `Device(config-vlan)# end`	Exits VLAN configuration mode and returns to privileged EXEC mode.

Configuring IPv6 Routing Proxy on an Interface

Before you begin

Follow these steps to enable IPv6 on an SVI:

```
Device# enable
Device# configure terminal
Device(config)# interface vlan vlan-id
Device(config-if)# no ipv6 redirects
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 address ipv6-address
```

To configure IPv6 routing proxy on an interface, follow this procedure:

Procedure

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet1/0/11`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**switchport access vlan** *vlan-id* **Example:** `Device(config)# switchport access vlan 15`	Assigns the port or range of ports into access ports.
Step 5	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Specifies which VLAN the interface belongs.
Step 6	**[no] ipv6 nd routing-proxy** **Example:** `Device(config-if)# ipv6 nd routing-proxy`	Specifies if the neighbor discovery suppress must operate in routing proxy mode.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring IPv6 DAD Proxy in VLAN Configuration Mode

Before you begin

- Follow these steps to enable IPv6 on an SVI:

```
Device# enable
Device# configure terminal
Device(config)# interface vlan vlan-id
```

```
Device(config-if)# no ipv6 redirects
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 address ipv6-address
```

- Attach a device tracking policy to the VLAN. For detailed steps, see the *Configuring Switch Integrated Security Features* chapter of the *Security Configuration Guide.*

To configure IPv6 DAD proxy in VLAN configuration mode, follow this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vlan configuration** *vlan-id* **Example:** `Device(config)# vlan configuration 15`	Enters the VLAN configuration mode. This mode allows you to name, set the state, disable, and shut down the VLAN or range of VLANs.
Step 4	**[no] ipv6 nd dad-proxy** **Example:** `Device(config-vlan)# ipv6 nd dad-proxy`	Specifies if the neighbor discovery suppress must operate in DAD proxy mode.
Step 5	**end** **Example:** `Device(config-vlan)# end`	Exits VLAN configuration mode and returns to privileged EXEC mode.

Configuring IPv6 DAD Proxy on an Interface

Before you begin

- Follow these steps to enable IPv6 on an SVI:

```
Device# enable
```

```
Device# configure terminal
Device(config)# interface vlan vlan-id
Device(config-if)# no ipv6 redirects
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 address ipv6-address
```

- Attach a device tracking policy to the layer 2 interface. For detailed steps, see the *Configuring Switch Integrated Security Features* chapter of the *Security Configuration Guide.*

To configure IPv6 DAD proxy on an interface, follow this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface GigabitEthernet1/0/11`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**switchport access vlan** *vlan-id* **Example:** `Device(config)# switchport access vlan 15`	Assigns the port or range of ports into access ports.
Step 5	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Specifies which VLAN the interface belongs.
Step 6	**[no] ipv6 nd dad-proxy** **Example:**	Specifies if the neighbor discovery suppress must operate in DAD proxy mode.

	Command or Action	Purpose
	`Device(config-if)# ipv6 nd dad-proxy`	
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying IPv6 Neighbor Discovery Proxy

Use the privileged EXEC or user EXEC commands in the table below to verify IPv6 neighbor discovery proxy information.

Table 7: Commands for Verifying IPv6 Neighbor Discovery Proxy

Commands	Description
show flooding-suppression	Displays flooding suppress policy (DAD proxy) configuration, and all the applied targets.
show ipv6 nd routing-proxy	Displays routing proxy default configuration, and all the applied targets .
show device-tracking policies	Displays device-tracking policy configuration, and all the applied targets.

Configuration Examples For IPv6 Neighbor Discovery Proxy

The following example shows the configuration of IPv6 routing proxy on a VLAN:

```
Device> enable
Device# configure terminal
Device(config)# vlan configuration  15
Device(config-vlan)# ipv6 nd routing-proxy
Device(config-vlan)# end
```

The following example shows the configuration of IPv6 DAD proxy on a VLAN:

```
Device> enable
Device# configure terminal
Device(config)# vlan configuration  15
Device(config-vlan)# ipv6 nd dad-proxy
Device(config-vlan)# end
```

The following example shows the output of the **show flooding-suppression** command in privileged EXEC mode:

```
Device# show flooding-suppression
```

```
Flooding suppress policy DAD_PROXY configuration:
  Suppressing  NDP
mode:DAD proxy- RFC6957
Policy DAD_PROXY is applied on the following targets:
Target               Type  Policy               Feature        Target range
vlan 15             VLAN  DAD_PROXY            Flooding Suppress vlan all
```

The following example shows the output of the **show ipv6 nd routing-proxy**command in privileged EXEC mode:

```
Device# show ipv6 nd  routing-proxy

Routing Proxy default configuration:
  Proxying NDP
Policy default is applied on the following targets:
Target               Type  Policy               Feature        Target range
vlan 15             VLAN  default              Routing Proxy  vlan all
```

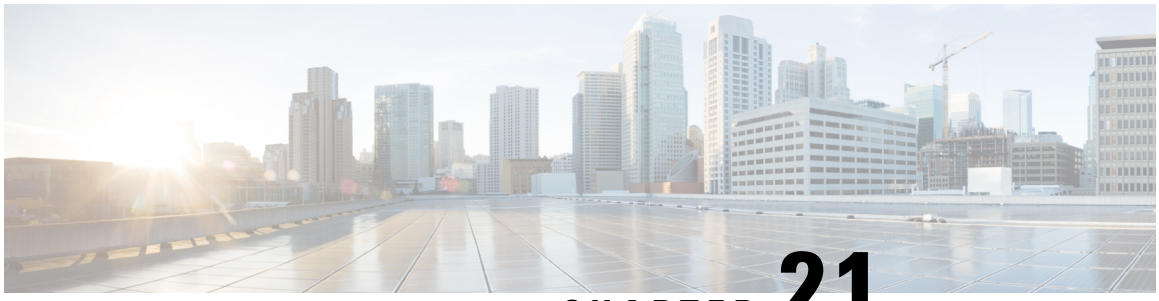
Feature History for IPv6 Neighbor Discovery Proxy

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 17.13.1	IPv6 Neighbor Discovery Proxy	Support for IPv6 Neighbor Discovery Proxy was introduced on all models of Cisco Catalyst 9600 Series Switches.

Use Cisco Feature Navigator to find information about platform and software image support.



CHAPTER 21

Troubleshooting IP Addressing Services

- Overview, on page 253
- Support Articles, on page 253
- Feedback Request, on page 254
- Disclaimer and Caution, on page 254

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
Operate and Troubleshoot DHCP Snooping on Catalyst 9000 Switches	This document describes how to operate and troubleshoot DHCP Snooping on Catalyst 9000 Series Switches.

Document	Description
Troubleshoot Slow Or Intermittent DHCP on Catalyst 9000 DHCP Relay Agents	This document describes how to troubleshoot slow Dynamic Host Configuration Protocol (DHCP) address allocation or intermittent DHCP address allocation failures on Catalyst 9000 Series Switches as DHCP relay agents.
Configure and Verify NAT on Catalyst 9000 Switches	This document describes how to configure and validate Network Address Translation (NAT) on the Catalyst 9000 Series Switches.

Feedback Request

Your input helps. A key aspect to improving these support documents is customer feedback. Note that these documents are owned and maintained by multiple teams within Cisco. If you find an issue specific to the document (unclear, confusing, information missing, etc):

- Provide feedback using the **Feedback** button located at the right panel of the corresponding article. The document owner will be notified, and will either update the article, or flag it for removal.
- Include information regarding the section, area, or issue you had with the document and what could be improved. Provide as much detail as possible.

Disclaimer and Caution

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.